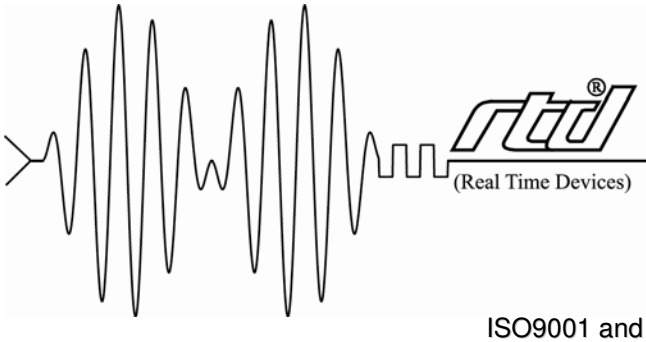
# DM7820/DM9820 User's Manual

***Versatile High Speed Digital I/O***



RTD Embedded Technologies, Inc.

*"Accessing the Analog World"®*

BDM-610010036
Rev B

ISO9001 and AS9100 Certified

# DM7820/DM9820 User's Manual

**RTD EMBEDDED TECHNOLOGIES, INC.**
103 Innovation Blvd
State College, PA 16803-0906

Phone: +1-814-234-8087
FAX: +1-814-234-5218

E-mail
sales@rtd.com
techsupport@rtd.com

Web Site
http://www.rtd.com

**Manual Revision History**

| | |
|---|---|
| Rev A | Initial Release |
| Rev B | Better DREQ description on page 40. Improved description of FIFO on page 12. Improved description of FIFOn_CON_STAT on page 41. Corrected PWM Period formula on page 55. Added DM9820 information. |

**Published by:**

RTD Embedded Technologies, Inc.
103 Innovation Boulevard
State College, PA 16803

Copyright 2009 by RTD Embedded Technologies, Inc.
All rights reserved

Specification and features described in this manual may change without notice.

The RTD Embedded Technologies Logo is a registered trademark of RTD Embedded Technologies. dspModule, cpuModule, and utilityModule are trademarks of RTD Embedded Technologies. PC/104, PC/104-Plus, PCI-104, PC/104-Express, and PCIe/104 are registered trademark of PC/104 Consortium. All other trademarks appearing in this document are the property of their respective owners.

# Table of Contents


Introduction ..... 1

Product Overview ..... 1
Board Features ..... 1
Digital I/O ..... 1
Deep FIFOs with DMA ..... 1
Pulse Width Modulators ..... 1
Incremental Encoders ..... 1
Advanced Interrupts ..... 2
Programmable Clocks ..... 2
82C54 Timer/Counters ..... 2
Physical Attributes ..... 2
Available Options ..... 2

Getting Technical Support ..... 3

Hardware Description ..... 4

Block Diagram ..... 4
Connector and Jumper Locations ..... 5
DM7820 ..... 5
DM9820 ..... 6
External I/O Connections ..... 6
Connector CN10 – Digital Input / Output ..... 6
Connector CN11 – Digital Input / Output ..... 7
PC/104 ISA Connectors –DM7820 ..... 8
PC/104-Express Bus Connectors–DM9820 ..... 8
PC/104-Plus PCI Connector ..... 8
PCI Configuration Options (DM7820 Only) ..... 8
Switch SW1 – PCI Slot Selector ..... 9
Jumper JP2 – Bus Master Control ..... 9
Solder Blob B1 – Force Three Master ..... 9

Board Installation ..... 10

Installing the Hardware ..... 10
Static Precautions ..... 10
Steps for Installing ..... 10
Installing Software ..... 11

Functional Overview ..... 12

Internal Architecture ..... 12
FIFOs ..... 12
Board Interrupts ..... 13
Advanced Triggering Examples ..... 14

Board Operation and Programming ..... 16

PCI Interface ..... 16
Device Memory ..... 16
Memory Map Overview ..... 16
Detailed Register Description ..... 26
System Block ..... 27
FPGA_VERSION ..... 27
SVN_VERSION ..... 27
BOARD_RESET ..... 27
BRD_STAT ..... 28

INT_ENABLE ... 28
INT_STATUS ... 29
Standard I/O ... 30
PORTx_OUTPUT ... 31
PORTx_INPUT ... 31
PORTx_TRISTATE ... 32
PORTx_MODE ... 32
PORTx_PERIPH_SEL_L ... 33
PORTx_PERIPH_SEL_H ... 33
STROBE_STATUS ... 34
82C54 Timer Counter Control ... 35
TC_ID ... 35
TC_INT ... 35
TC_xy_CONTROL ... 36
FIFO Channel n ... 37
FIFOn_ID ... 37
FIFOn_INT ... 37
FIFOn_IN_CLK ... 38
FIFOn_OUT_CLK ... 39
FIFOn_IN_DATA_DREQ ... 40
FIFOn_CON_STAT ... 41
FIFOn_RW_PORT ... 41
Programmable Clock n ... 41
PROGCLKn_ID ... 42
PROGCLKn_MODE ... 42
PRGCLKn_CLK ... 42
PRGCLKn_START_STOP ... 43
PROGCLKn_PERIOD ... 45
Advanced Interrupt n ... 46
ADVINTn_ID ... 46
ADVINTn_INT_MODE ... 46
ADVINTn_CLK ... 47
ADVINTn_PORTx_MASK ... 47
ADVINTn_PORTx_CMP ... 48
ADVINTn_PORTx_CAPT ... 48
Dual Incremental Encoder n ... 48
INCENCn_ID ... 49
INCENCn_INT ... 50
INCENCn_CLK ... 51
INCENCn_MODE ... 51
INCENCn_VALUEy ... 52
Quad Pulse Width Modulator n ... 53
PWMn_ID ... 54
PWMn_MODE ... 54
PWMn_CLK ... 54
PWMn_PERIOD ... 55
PWMn_WIDTHx ... 55
82C54 Timer Counter n ... 55
DESCRIPTION OF OPERATION ... 56
Control Word and Count Value Program ... 56
Mode definition ... 57
Mode 0 ... 57
Mode 1 ... 58
Mode 2 ... 58
Mode 3 ... 58
Mode 4 ... 59

Mode 5 ..... 60
Reading Counter Values ..... 62
Direct reading ..... 62
Counter latching ..... 62
Read Back Command Operation ..... 63
PLX Registers ..... 66
Memory Map Overview ..... 66
DMA Register Description ..... 67
DMAMODEn ..... 67
DMAPADRn ..... 69
DMALAPADRn ..... 69
DMASIZn ..... 70
DMAPRn ..... 70
DMACSRn ..... 70
DMAARB ..... 71
DMATHR ..... 71
DMADAn ..... 73
INTCSR ..... 73
Additional Information ..... 77
PLX PCI9056 ..... 77
82C54 Timer/Counter Programming ..... 77
Interrupt Programming ..... 77
DC Characteristics ..... 78
Absolute Maximum Ratings ..... 78
DC Input / Output Levels ..... 78
Limited Warranty ..... 79

# Table of Figures

Figure 1: DM7820/DM9820 Block Diagram ..... 4

Figure 2: Digital I/O Block Diagram ..... 12

Figure 3: Interrupt Diagram ..... 14

Figure 4: Digital I/O Block Diagram ..... 31

Figure 5: Incremental Encoder Signals ..... 49

Figure 6: PWM Output ..... 53

Figure 7:  Counter latching executed for counter #1 (Read/Load 2-byte setting) ..... 63

# Table of Tables

Table 1: CN10 Pin Assignments ..... 6
Table 2: CN11 Pin Assignments ..... 7
Table 3: PCI Configuration Registers ..... 16
Table 4: DM7820/DM9820 Memory Map ..... 17
Table 5: Peripheral Outputs ..... 33
Table 6: Incremental Encoder Inputs ..... 49
Table 7: Select Counter SC[1:0]: Selection of set counter ..... 56
Table 8: Read/Load RL[1:0]: Count Value Reading/Loading format setting ..... 56
Table 9: Mode M[2:0]: Operation waveform mode setting ..... 56
Table 10: BCD: Operation count mode setting ..... 57
Table 11: PLX DM7820/DM9820 Memory Map ..... 66
Table 12: DMA Threshold Nybble Values ..... 72

# Introduction

## *Product Overview*

The DM7820/DM9820 is designed to provide high speed digital I/O for PC/104-Plus Systems. It interfaces with the PCI bus and uses large FIFOs and DMA transfers to allow for efficient data management. Several peripherals, including Pulse Width Modulators, Incremental Encoders, and Programmable Clocks are also provided.

## *Board Features*

### Digital I/O

48 Diode protected I/O lines
24 mA source and sink current
Compatible with DMR and DOP expansion boards

### Deep FIFOs with DMA

Two 2M-Word FIFOs
Each FIFO is attached to a separate DMA channel
25 MHz bursted throughput
12.5 MHZ continuous throughput
FIFO can be looped

### Pulse Width Modulators

Eight PWM outputs
Single-ended or Differential Outputs
16-bit resolution
Separate period and width clocks provide full resolution at low duty cycles
Optional Interrupt generations

### Incremental Encoders

Four Incremental Encoder channels
Single-ended or Pseudo-differential Inputs
Variable frequency input filtering
Max input speed of 40ns per transition
16-bit resolution
Two channels can be combined for 32-bit resolution
Connect to FIFO for position sampling

### Advanced Interrupts

Two Advanced Interrupt Modules
Interrupt on Match, Change, or Strobe
All 48 bits are captured when the interrupt is generated
Any combination of the 48 bits can be monitored

### Programmable Clocks

Four programmable clocks
Maximum frequency of 25 MHz
Can be started and stopped by an interrupt or another clock
Continuous or One-Shot Operation
Can be cascaded

### 82C54 Timer/Counters

Six Timer/Counter Channels
Fully programmable
Input clock and gate driven from internal or external source
10 MHz maximum input

### Physical Attributes

Size: 3.6”L x 3.8”W x 0.6”H (90mm L x 96mm W x 15mm H)
Weight: 0.22 lbs (0.10 Kg)
Operating Temperature: -40$^{o}$ C to +85$^{o}$ C
Storage Temperature: -55$^{o}$ C to +125$^{o}$ C
Power Requirements:
Typical: 1.5 W @ +5 VDC

## *Available Options*

The DM7820/DM9820 is a modular design. Custom feature sets are available. Please contact RTD Embedded Technologies for more information on custom boards.

# Getting Technical Support

For help with this product, or any other product made by RTD, you can contact RTD Embedded Technologies via the following methods:

Phone: +1-814-234-8087

E-Mail: techsupport@rtd.com

Be sure to check the RTD web site (http://www.rtd.com) frequently for product updates, including newer versions of the board manual and application software.

# Hardware Description

## *Block Diagram*

Below is a block diagram of the DM7820/DM9820. Primary board components are in bold, while external I/O connections and jumpers are italicized.

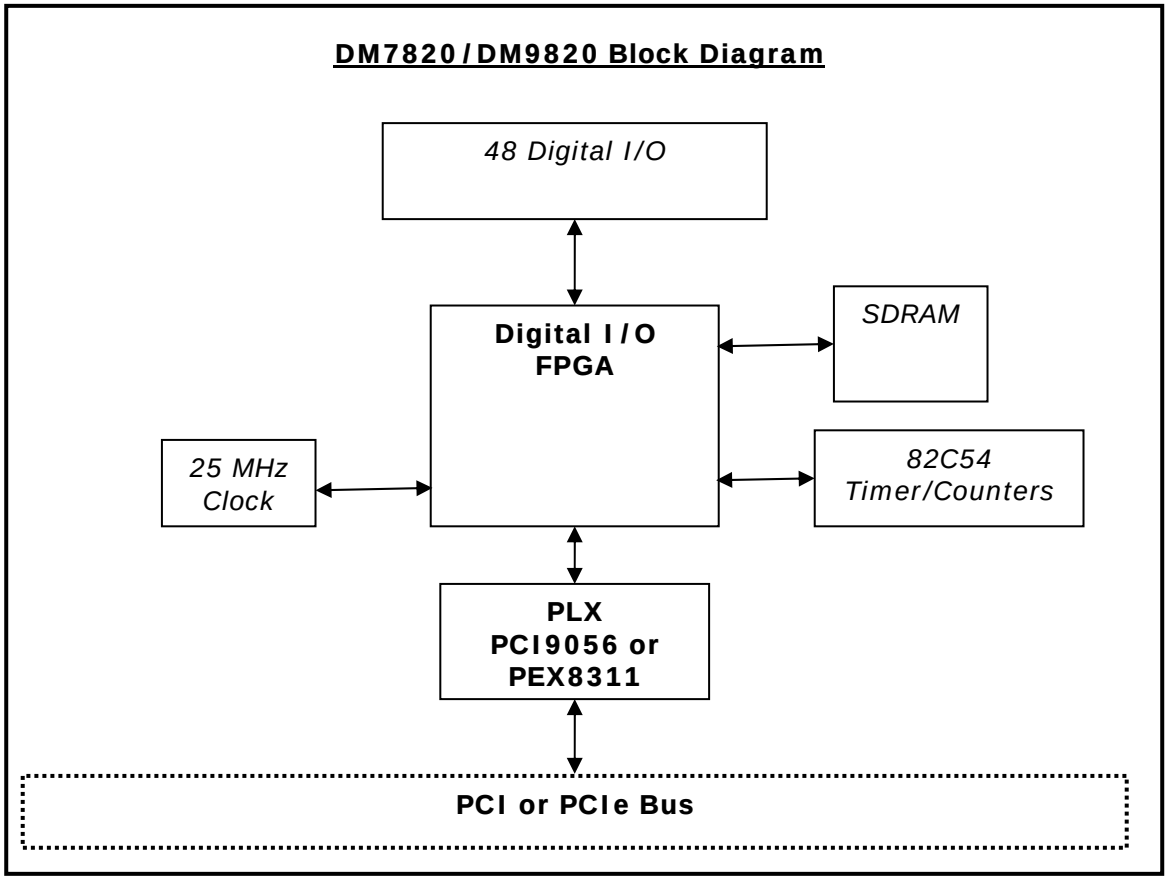


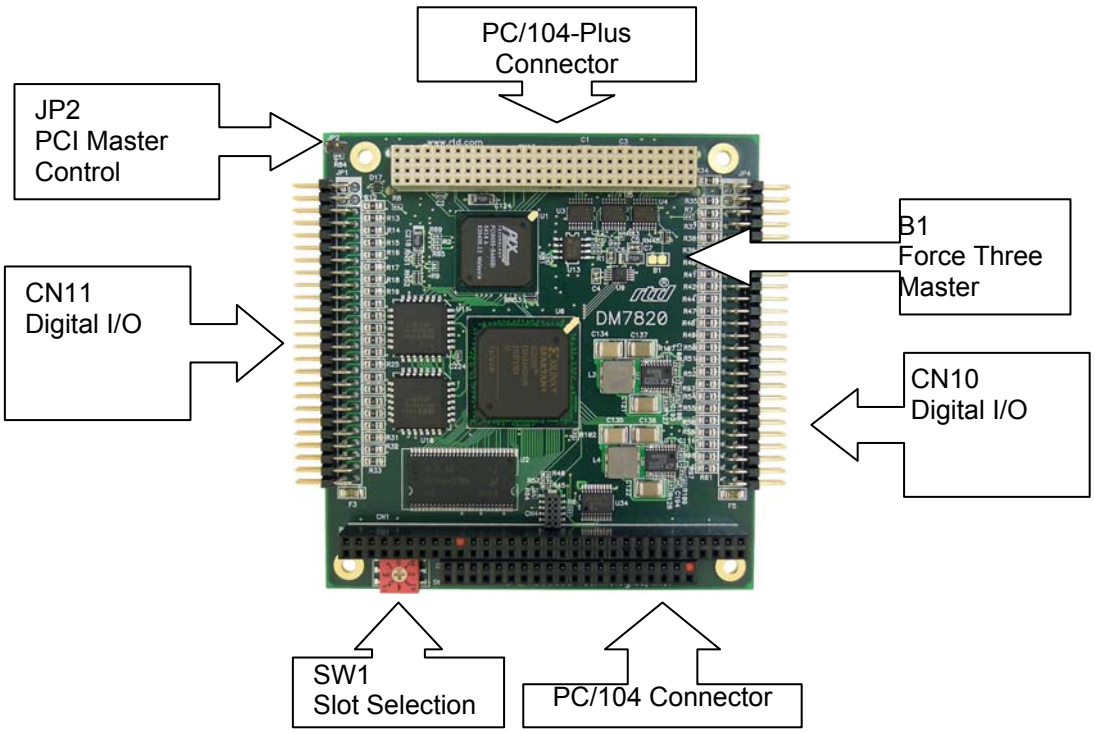

**Figure 1: DM7820/DM9820 Block Diagram**

# *Connector and Jumper Locations*

## DM7820

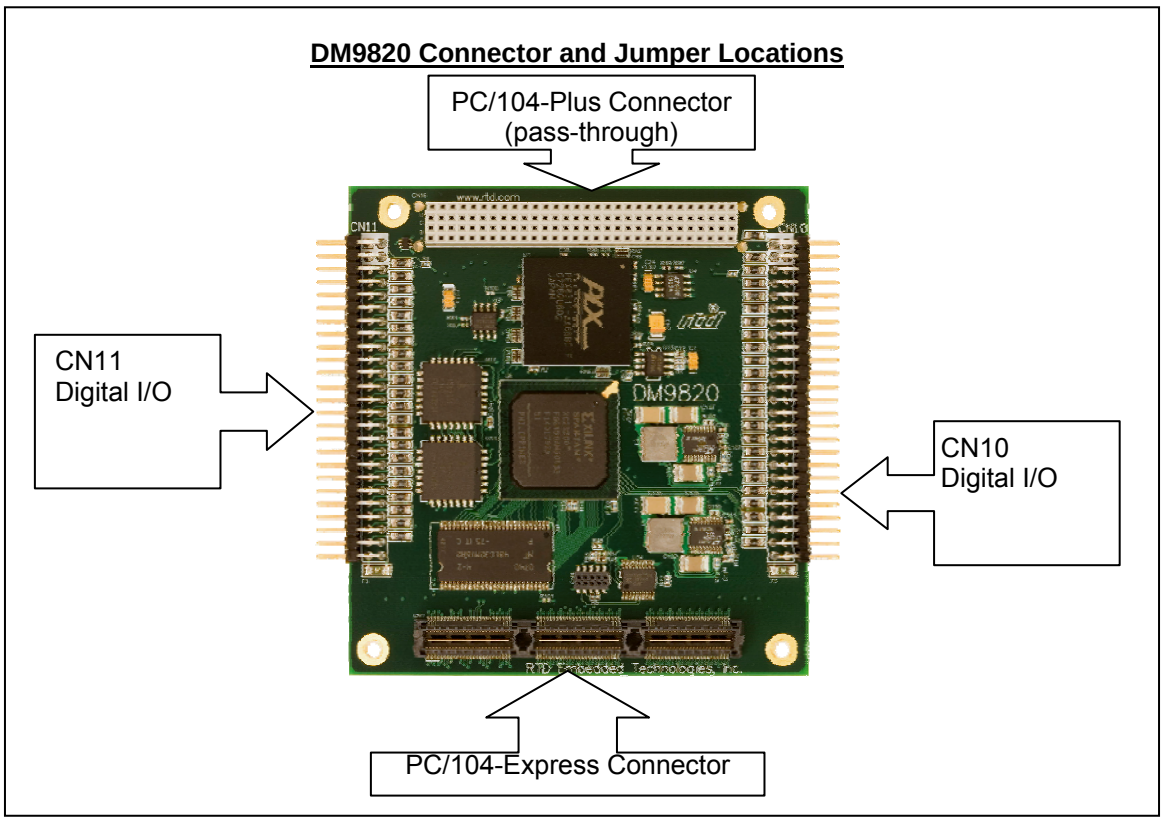
The following diagram shows the location of all connectors and jumpers on the DM7820. For a description of each jumper and connector, refer to the following sections.

**DM7820 Connector and Jumper Locations**



## DM9820

The following diagram shows the location of all connectors and jumpers on the DM9820. For a description of each jumper and connector, refer to the following sections.




# *External I/O Connections*

The following sections describe the external I/O connections of the DM7820/DM9820.

## Connector CN10 – Digital Input / Output

Connector CN10 provides 24 digital input/output lines, along with a +5V pin and ground pins. The pin assignments for CN10 are shown in Table 1.

**Note:** Pin 1 can be identified by a square solder pad. Pins 2 – 50 have round solder pads.

**Table 1: CN10 Pin Assignments**

| Signal | Pin | Pin | Signal |
|---|---|---|---|
| P2[7] | 1 | 2 | Strobe2 |
| P2[6] | 3 | 4 | GND |
| P2[5] | 5 | 6 | GND |
| P2[4] | 7 | 8 | GND |
| P2[3] | 9 | 10 | GND |
| P2[2] | 11 | 12 | GND |

**Table 1: CN10 Pin Assignments**

| Signal | Pin | Pin | Signal |
|---|---|---|---|
| P2[1] | 13 | 14 | GND |
| P2[0] | 15 | 16 | GND |
| P0[15] | 17 | 18 | GND |
| P0[14] | 19 | 20 | GND |
| P0[13] | 21 | 22 | GND |
| P0[12] | 23 | 24 | GND |
| P0[11] | 25 | 26 | GND |
| P0[10] | 27 | 28 | GND |
| P0[9] | 29 | 30 | GND |
| P0[8] | 31 | 32 | GND |
| P0[7] | 33 | 34 | GND |
| P0[6] | 35 | 36 | GND |
| P0[5] | 37 | 38 | GND |
| P0[4] | 39 | 40 | GND |
| P0[3] | 41 | 42 | GND |
| P0[2] | 43 | 44 | GND |
| P0[1] | 45 | 46 | GND |
| P0[0] | 47 | 48 | GND |
| +5V, 2A max | 49 | 50 | GND |

See Table 5 and Table 6 for peripheral pin assignments.

## Connector CN11 – Digital Input / Output

Connector CN11 provides 24 digital input/output lines, along with a +5V pin and ground pins. The pin assignments for CN11 are shown in Table 2.

**Note:** Pin 1 can be identified by a square solder pad. Pins 2 – 50 have round solder pads.

**Table 2: CN11 Pin Assignments**

| Signal | Pin | Pin | Signal |
|---|---|---|---|
| P2[15] | 1 | 2 | Strobe1 |
| P2[14] | 3 | 4 | GND |
| P2[13] | 5 | 6 | GND |
| P2[12] | 7 | 8 | GND |
| P2[11] | 9 | 10 | GND |
| P2[10] | 11 | 12 | GND |
| P2[9] | 13 | 14 | GND |
| P2[8] | 15 | 16 | GND |
| P1[15] | 17 | 18 | GND |
| P1[14] | 19 | 20 | GND |
| P1[13] | 21 | 22 | GND |
| P1[12] | 23 | 24 | GND |
| P1[11] | 25 | 26 | GND |
| P1[10] | 27 | 28 | GND |
| P1[9] | 29 | 30 | GND |
| P1[8] | 31 | 32 | GND |

**Table 2: CN11 Pin Assignments**

| Signal | Pin | Pin | Signal |
|---|---|---|---|
| P1[7] | 33 | 34 | GND |
| P1[6] | 35 | 36 | GND |
| P1[5] | 37 | 38 | GND |
| P1[4] | 39 | 40 | GND |
| P1[3] | 41 | 42 | GND |
| P1[2] | 43 | 44 | GND |
| P1[1] | 45 | 46 | GND |
| P1[0] | 47 | 48 | GND |
| +5V, 2A max | 49 | 50 | GND |

See Table 5 and Table 6 for peripheral pin assignments.

# *PC/104 ISA Connectors –DM7820*

The PC/104 connectors carry the signals of the PC/104-*Plus* ISA bus. Refer to PC/104-*Plus* Specification Revision 1.0 for the pinout of this connector. This is a pass-through connector. The DM7820 connects to the power and ground pins only, and does not use any of the signals.

# *PC/104-Express Bus Connectors–DM9820*

The PC/104-Express connectors provide the PCI Express bus connections. CN1 is on the top, and CN2 is on the bottom. Refer to the PC/104-Express™ Specification Revision 1.0 for the pinout of these connectors.

The DM9820 connects to one of the PCIe x1 links on the PCIe bus connector, and passes through the x16 link. It will automatically detect the direction to the host, so it can be stacked above or below the CPU.

# *PC/104-Plus PCI Connector*

The PC/104-*Plus* connector carries the signals of the PC/104-*Plus* PCI bus. Refer to PC/104-*Plus*™ Specification for the pinout of this connector. The DM9820 connects to the power and ground pins only, and does not use any of the signals. The DM7820 uses this connector for communication with the CPU.

## PCI Configuration Options (DM7820 Only)

To install the DM7820 into the stack, the PCI Slot Number must be configured correctly. This is done by the PCI Slot Selector located at SW1.

There are four possible PCI Slot Numbers (0 – 3). Each PCI device (PC/104-*Plus* or PCI-104) must a use a different slot number. The slot number is related to the position of the board in the stack. Slot 0 represents the PCI device closest to the CPU. Slot 3 represents the PCI devices farthest away from the CPU.

**Note:** In a PC/104-*Plus* or PCI-104 system, all PCI devices should be located on one side of the CPU board (above or below the add-on cards). The CPU should not be located between two PCI devices.

## Switch SW1 – PCI Slot Selector

When the PC/104-*Plus* Specification was first introduced, it only allowed for three PCI add-on cards to be bus masters. Version 2.0 of the PC/104-*Plus* specification was released in November 2003. This version of the specification (which the DM7820 is designed for) adds support for all 4 PCI slots to be bus masters.

There are two methods for compatibility with CPUs designed for the older PC/104-*Plus* Specification. One method is to use slot positions 4-7 instead of the usual 0-3. The second is to short solder jumper B1.

The PCI Slot Number can be configured as follows:

| Switch Position | PCI Slot Number | Compatibility | Master |
|---|---|---|---|
| 0 | Slot 0 (closest to CPU) | 4 | yes |
| 1 | Slot 1 | 4 | yes |
| 2 | Slot 2 | 4 | yes |
| 3 | Slot 3 | 4 | yes |
| 4 | Slot 0 (closest to CPU) | 3 | yes |
| 5 | Slot 1 | 3 | yes |
| 6 | Slot 2 | 3 | If JP2 |
| 7 | Slot 3 | 3 | If JP2 |

## Jumper JP2 – Bus Master Control

Install JP2 to enable bus mastering when in Slot 2 or Slot 3 in three bus master mode.

## Solder Blob B1 – Force Three Master

The DM7820 offers a configuration solder blob at location B1. If this solder blob is open (the default), the board supports bus mastering in all 4 PCI slots when SW1 is in position 0-3. If it is closed, the board will work in a 3 bus master configuration. If B1 is closed, SW1 positions 0-3 will be identical to positions 4-7.

**Note:** The DM7820 comes with solder blob B1 open by default. This should be compatible with most PC/104-*Plus* CPUs. There is no need to change this blob unless you are having compatibility problems with your specific CPU.

# Board Installation

## *Installing the Hardware*

The DM7820 can be installed into a PC/104-*Plus* or PCI-104 stack. It can be located above or below the CPU, as long as all PCI add-on cards are on the same side of the CPU.

The DM9820 can be installed into a PC/104-Express or PCIe/104 stack. It can be located above or below the CPU.

### Static Precautions

Keep your board in its antistatic bag until you are ready to install it into your system! When removing it from the bag, hold the board at the edges and do not touch the components or connectors. Handle the board in an antistatic environment and use a grounded workbench for testing and handling of your hardware.

### Steps for Installing

1. Shut down the PC/104 system and unplug the power cord.
2. Ground yourself with an anti-static strap.
3. Set the PCI Slot Selector as described in the previous chapter.
4. If any other PCI add-on cards are to be included in the stack, be sure that their PCI slot numbers are configured correctly (Slot 0 for the board closest to the CPU, Slot 1 for the next board, etc).
5. Line up the pins of the DM7820/DM9820's connectors with the corresponding bus connectors of the stack. Make sure that both connectors are lined up.
6. Apply pressure to both bus connectors and gently press the board onto the stack. The board should slide into the matching bus connectors. Do not attempt to force the board, as this can lead to bent/broken pins.
7. Attach any cables to the DM7820/DM9820
8. If any boards are to be stacked above the DM7820/DM9820, install them.
9. Attach any necessary cables to the PC/104-*Plus* stack.
10. Re-connect the power cord and apply power to the stack.
11. Boot the system and verify that all of the hardware is working properly.

**Note:** If multiple PCI devices are configured to use the same PCI slot number, the system will not boot.

# *Installing Software*

Drivers are required to use the DM7820/DM9820. They are provided, along with example programs, on the CD that ships with the board, and are also available from the RTD website (www.rtd.com). For further information on installing the drivers, review README.TXT in the driver archive file.

# Functional Overview

## *Internal Architecture*

A diagram of the standard I/O is shown in Figure 4. Each digital I/O pin can be an input, output, or peripheral output. The peripheral outputs are the Pulse Width Modulators, FIFO, Timer/Counters, etc.

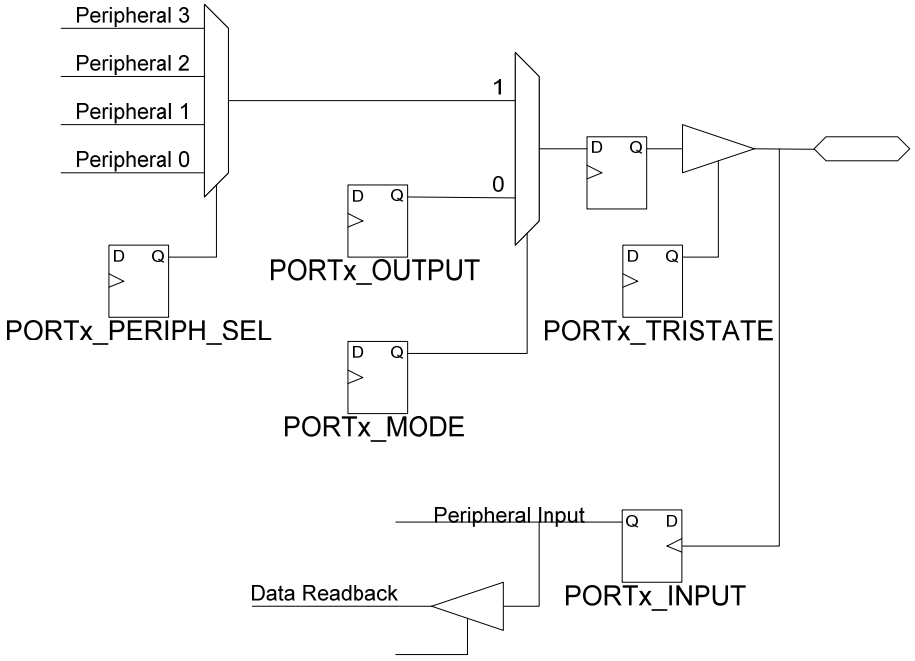



**Figure 2: Digital I/O Block Diagram**

## *FIFOs*

The DM7820/DM9820 provides two FIFOs to buffer data going into and out of the board. Each FIFO is 16-bit wide and 2,097,661 Words deep. The input strobe, output strobe, and data input for each FIFO can be individually selected. The output data is made available to the peripheral outputs, and also the PCI interface.

Each FIFO is attached to a DMA Channel in the PLX chip. FIFO0 is attached to DMA0, and FIFO1 is attached to DMA1.

FIFO0 can have its input data attached to its output data. In this case, the same data is repeated forever. This is useful for some types of pattern generation.

Internally, the FIFO system consists of a single 8MB SDRAM device, with 255-word input and output buffers for each channel. When data is available in the input buffer, it is moved into the area of SDRAM device for that channel. When data is in the SDRAM device, and there is room available in the output buffer, data is moved to the output buffer. All of the internal data movement is handled automatically. Greatest data efficiency is achieved when there is at least 128 words of data in the FIFO.

The FIFO also provides “Write Request” and “Read Request” signals. For these signals, the internal buffers are monitored to signal when data can be sent into, and read from the FIFO. The “Write Request” is asserted when there is at least 256 words of space available in the FIFO, and negated when there is less than 128 words available. The “Read Request” is asserted when at least 256 words of data is in the FIFO, and negated when there is less than 128 words of data. Using these signals guarantees a burst of at least 128 words, which provides for efficient communication over the PCI bus, and robustly guards against over-run and under-run conditions. However, it does not allow for the FIFO to be completely filled of emptied.

There is a total of 45 M-words per second of available bandwidth for the entire FIFO system. This bandwidth is allocated between all input and output sources. This is assuming that at least 256 Words stay in the FIFO at all times to maximize bursting (i.e., the Read Request and Write Request are used for DREQ). If only one word is available in the FIFO (i.e. Not Empty is used as for DREQ) the available bandwidth drops to 3.75 M-words per second. When a FIFO is looped, the data must be read and written. The table below shows examples of configurations and their maximum data rate. Note that for uniform sampling (samples are taken at uniform sampling intervals) the data rate must be an integer divisor of the 25 MHz overall clock.

| Description | Max Data Rate |
|---|---|
| One FIFO in use, burst capture only | 25 MHz |
| One FIFO in use, burst output only | 25 MHz |
| One FIFO in use, continuous capture or output | 12.5 MHz |
| Two FIFOs in use, continuous capture/output | 12.5 MHz + 6.25 MHz |
| Two FIFOs in use, continuous capture/output, non-uniform sampling | 11 MHz + 11 MHz |

# *Board Interrupts*

There are three levels of interrupt sources for this board: the interrupt sources generated in the PLX chip, the interrupt sources generated by the modules in the Control Block, and the interrupt sources within the modules. Each level must be enabled in the previous level. Figure 3 shows a block diagram of the interrupt sources. Note that there are some other sources in the PLX bridge chip; consult the datasheet for more details.

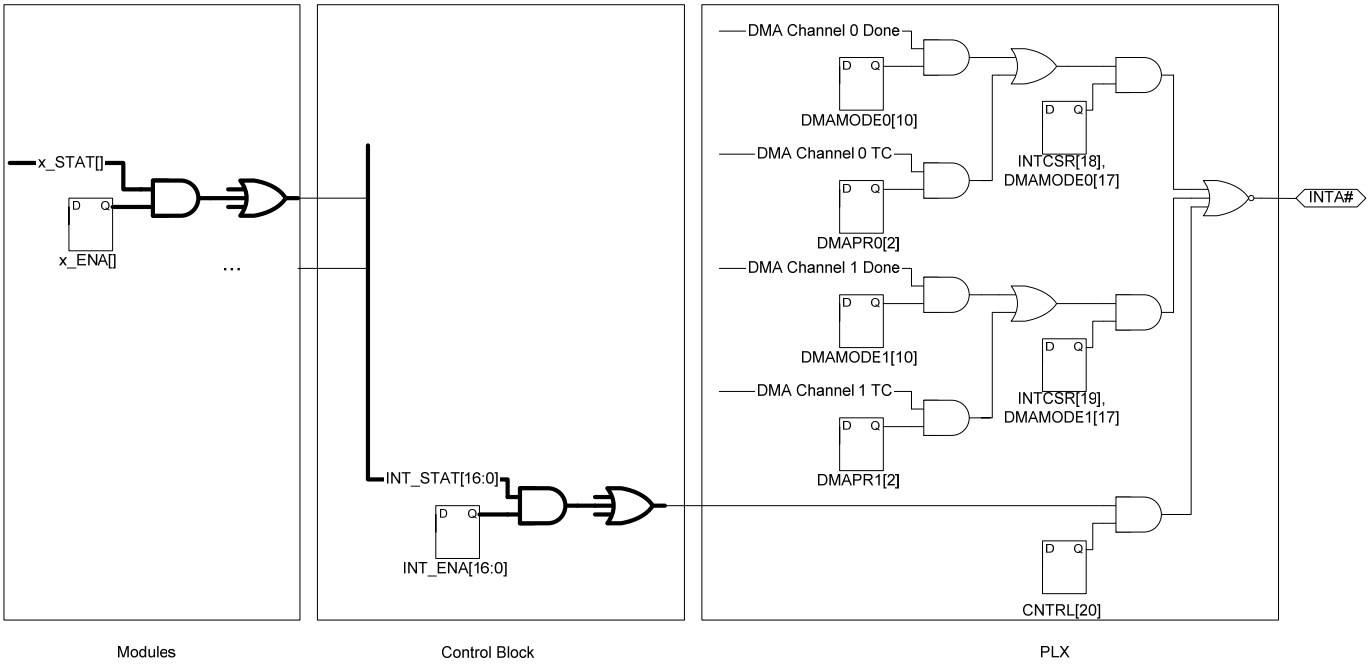



**Figure 3: Interrupt Diagram**

# *Advanced Triggering Examples*

The modules on the DM7820/DM9820 can be combined to generate a broad range of complex sampling scenarios. The following example shows how to use the Advanced Interrupt and 4 counters to capture N words before and M words after an event. Programmable Clock 0 is the sample clock, and is used to clock data into the FIFO. It is started after all of the other Programmable clocks are initialized. As soon as it starts, Programmable Clock 1 starts counting samples to be captured before the triggering event. This is also known as “pre-fill.” When it expires, it starts Programmable Clock 2, which removes samples from the FIFO at the same rate that they are stored, keeping a constant number of samples in the FIFO. When the triggering event happens, Programmable Clock 2 is stopped, and the FIFO begins to fill. Also, the triggering event starts Programmable Clock 3, which counts the number of samples to be captured after the triggering event. When Programmable Clock 3 expires, it stops Programmable Clock 0, and data collection ends. The triggering event can also generate an interrupt that changes the FIFO output to PCI Read, and start DMA transfers. This allows the data to be moved to system memory before data collection has ended.

- AdvInt0
    - Set to event desired
    - During the Interrupt Service Routine…
        - Change FIFO output clock to PCI Read
        - Start DMA transfers
- Prog Clock 0 - Sample Input Clock
    - Period = sample period

    - Master Clock = any
    - Start Event = always
    - Stop Event = Prog Clock 3
    - Continuous operation
- Prog Clock 1 - Pre-capture clock
    - Period = N (samples before event)
    - Master Clock = Prog Clock 0
    - Start Event = always
    - One-shot
- Prog Clock 2 - Sample output clock
    - Period = same as Prog Clock 0
    - Master Clock = same as Prog Clock 0
    - Start Event = Prog Clock 1
    - Stop Event = AdvInt 0
    - Continuous
- Prog Clock 3 - Post-Capture clock
    - Period = M (samples after event)
    - Master Clock = Prog Clock 0
    - Start Event = AdvInt0
    - One-Shot
- FIFO 0
    - Data In = any
    - Data in clock = Prog Clock 0
    - Data out clock = Prog Clock 2 (before AdvInt0)
    - Data out clock = PCI Read (after Adv Int0)
    - DReq0 = Read Ready

# Board Operation and Programming

## PCI Interface

This board attaches to the PCI bus using a PLX PCI9056. The PCI9056 is operating in "C Mode." Most of the registers in the PLX chip are automatically programmed at power up by the on-board EEPROM or by the system BIOS. The only PLX registers that the user needs to access are the DMA registers found on page 66, and the Configuration Registers found in Table 3 below.

For more information on the PLX PCI9056 bridge chip, contact PLX Technologies, (www.plxtech.com).

**Table 3: PCI Configuration Registers**

| PCI Config. Register Address (Hex) | Register Name | | | |
|---|---|---|---|---|
| **0x00** | PCI Device ID = 0x7820 | | PCI Vendor ID = 0x1435 | |
| **0x04** | PCI Status | | PCI Command | |
| **0x08** | PCI Class Code = 0x118000 | | | PCI Rev. ID = 0x00 |
| **0x0C** | PCI BIST | PCI Header Type | Bus Latency Timer | Cache Line Size |
| **0x10** | PCI Base Address Register 0: Memory Access to PLX9056 Registers | | | |
| **0x14** | PCI Base Address Register 1: I/O Access to PLX9056 Registers | | | |
| **0x18** | PCI Base Address Register 2: Memory Access to Digital I/O Registers | | | |
| **0x1C** | PCI Base Address Register 3: Reserved | | | |
| **0x20** | Reserved | | | |
| **0x24** | Reserved | | | |
| **0x28** | Reserved | | | |
| **0x2C** | PCI Subsystem ID = 0x9056 | | PCI Subsystem Vendor = 0x10B5 | |
| **0x30** | Reserved | | | |
| **0x34** | Reserved | | | Reserved |
| **0x38** | Reserved | | | |
| **0x3C** | PCI Max Latency | PCI Min Grant | PCI Interrupt Pin | PCI Interrupt Line |

## Device Memory

The DM7820/DM9820 is a memory mapped device. The address for the memory mapped registers can be found in Base Address Register 2. Generally, the registers are 16 bits wide. However, they can be read and written as 8, 16, or 32 bits. (There are a few exceptions as noted in the memory map.)

### Memory Map Overview

Table 4 shows the memory map of the DM7820/DM9820 digital I/O registers. These are found at the offset from BAR2.

**Table 4: DM7820/DM9820 Memory Map**

| Offset (Hex) | Register Name | Register Function |
|---|---|---|
| | | **Board Control** |
| **0x0000** | FPGA_VERSION | b[15:8] Type ID<br>b[7:0] Version |
| **0x0002** | SVN_VERSION | b[15:0] Extended Version |
| **0x0004** | BOARD_RESET | Write 0xA5A5 to reset board |
| **0x0006** | Reserved | |
| **0x0008** | BRD_STAT | b[15:1] Reserved<br>b0 MSTR – 0 = PCI Master Capable, 1 = Not PCI Master Capable (Read Only) |
| **0x0010** | INT_ENABLE | b[15:0] Interrupt Enable – Set to '1' to enable a specific interrupt |
| **0x0012** | INT_STAT | b [15:0] Interrupt Status – Reading a '1' indicates interrupt condition has occurred. Write a '1' to clear an interrupt bit. |
| **0x0014–0x003E** | Reserved | |
| | | **Standard I/O** |
| **0x0040** | PORT0_OUTPUT | b [15:0] Value to port 0 when it is an output. |
| **0x0042** | PORT0_INPUT | b [15:0] Read only value from Port 0. |
| **0x0044** | PORT0_TRISTATE | b [15:0] '1' for output, '0' for input. |
| **0x0046** | PORT0_MODE | b[15:0] '1' for peripheral output, '0' for digital I/O |
| **0x0048** | PORT1_OUTPUT | b [15:0] Value to port 1 when it is an output. |
| **0x004A** | PORT1_INPUT | b [15:0] Read only value from Port 1. |
| **0x004C** | PORT1_TRISTATE | b [15:0] '1' for output, '0' for input. |
| **0x004E** | PORT1_MODE | b[15:0] '1' for peripheral output, '0' for digital I/O |
| **0x0050** | PORT2_OUTPUT | b [15:0] Value to port 2 when it is an output. |
| **0x0052** | PORT2_INPUT | b [15:0] Read only value from Port 2. |
| **0x0054** | PORT2_TRISTATE | b [15:0] '1' for output, '0' for input. |
| **0x0056** | PORT2_MODE | b[15:0] '1' for peripheral output, '0' for digital I/O |
| **0x0058** | STROBE_STATUS | b [9] STR2_TRI – '0' = strobe2 is input, '1' = srobe2 is output.<br>b [8] STR1_TRI – '0' = strobe1 is input, '1' = srobe1 is output.<br>b [5] STR2_OUT – Value for strobe2 when an output.<br>b [4] STR1_OUT – Value for strobe1 when an output.<br>b [1] STR2_IN – Current value of Strobe2.<br>b [0] STR1_IN – Current value of Strobe1. |
| **0x005A–0x005E** | Reserved | |

**Table 4: DM7820/DM9820 Memory Map**

| Offset (Hex) | Register Name | Register Function |
|---|---|---|
| **0x0060** | PORT0_PERIPH_SEL_L | b[15:14]Port0[7]_Periph_Select<br>b[13:12]Port0[6]_Periph_Select<br>b[11:10]Port0[5]_Periph_Select<br>b[9:8] Port0[4]_Periph_Select<br>b[7:6] Port0[3]_Periph_Select<br>b[5:4] Port0[2]_Periph_Select<br>b[3:2] Port0[2]_Periph_Select<br>b[1:0] Port0[0]_Periph_Select |
| **0x0062** | PORT0_PERIPH_SEL_H | b[15:14]Port0[15]_Periph_Select<br>b[13:12]Port0[14]_Periph_Select<br>b[11:10]Port0[13]_Periph_Select<br>b[9:8] Port0[12]_Periph_Select<br>b[7:6] Port0[11]_Periph_Select<br>b[5:4] Port0[10]_Periph_Select<br>b[3:2] Port0[9]_Periph_Select<br>b[1:0] Port0[8]_Periph_Select |
| **0x0064** | PORT1_PERIPH_SEL_L | |
| **0x0066** | PORT1_PERIPH_SEL_H | |
| **0x0068** | PORT2_PERIPH_SEL_L | |
| **0x006A** | PORT2_PERIPH_SEL_H | |
| **0x0070–0x007E** | Reserved | |
| **82C54 Timer Counter Control** | | |
| **0x0080** | TC_ID | b[15:0] ID Register – equals 0x1001 |
| **0x0082** | TC_INT | b[15:14]Reserved<br>b [13:8] Interrupt Status – ‘1’ = Interrupt condition has occurred. Write ‘1’ to clear. Interrupts are asserted on the positive edge of the clock.<br>b[7:6] Reserved<br>b[5:0] Interrupt Enable – ‘1’ = Interrupt is enabled, ‘0’ = disabled<br>Interrupt source are:<br>5 TC B2<br>4 TC B1<br>3 TC B0<br>2 TC A2<br>1 TC A1<br>0 TC A0 |
| **0x0084** | TC_A0_CONTROL | b[15:13]Reserved<br>b[12:8] Gate Select<br>31-16 = Port 2 [15-0]<br>15-2 = Clock_Bus [15-2]<br>1 = ‘1’<br>0 = ‘0’<br>b[7:4] Reserved<br>b[3:0] Clock Select<br>15-2 = Clock_Bus [15-2]<br>1 = reserved<br>0 = 5 MHz |
| **0x0086** | TC_A1_CONTROL | |
| **0x0088** | TC_A2_CONTROL | |
| **0x008A** | TC_B0_CONTROL | |
| **0x008C** | TC_B1_CONTROL | |
| **0x008E** | TC_B2_CONTROL | |
| **FIFO Channel 0** | | |

**Table 4: DM7820/DM9820 Memory Map**

| Offset (Hex) | Register Name | Register Function |
|---|---|---|
| **0x00C0** | FIFO0_ID | b[15:0] ID Register = 0x2011 |
| **0x00C2** | FIFO0_INT | b [15:8] Interrupt Status – '1' = Interrupt condition has occurred. Write '1' to clear.<br>b[7:0] Interrupt Enable – '1' = Interrupt is enabled, '0' = disabled<br>Interrupt source are:<br>7 Reserved<br>6 Reserved<br>5 Underflow<br>4 Overflow<br>3 Empty<br>2 Full<br>1 Write Request<br>0 Read Request |
| **0x00C4** | FIFO0_IN_CLK | b[15:5] Reserved<br>b[4:0] Input Clock Select<br>31 = PCI Write<br>30 = PCI Read<br>29-16 = Interrupts[13-0]<br>15-0 = Clock_Bus [15-0] |
| **0x00C6** | FIFO0_OUT_CLK | b[15:5] Reserved<br>b[4:0] Input Clock Select<br>31 = PCI Write<br>30 = PCI Read<br>29-16 = Interrupts[13-0]<br>15-0 = Clock_Bus [15-0] |
| **0x00C8** | FIFO0_IN_DATA_DREQ | b[15:10]Reserved<br>b[9:8] DREQ0 Source<br>3 = Not Full<br>2 = Write Request<br>1 = Not Empty<br>0 = Read Request<br>b[7:4] Reserved<br>b[3:0] Input Data Select<br>3 = FIFO0 Output<br>2 = Port 2<br>1 = Port 0<br>0 = PCI Data |
| **0x00CA** | FIFO0_CON_STAT | b[15:10]Reserved<br>b[9] Write Request (non-sticky)<br>b[8] Read Request (non-sticky)<br>b[7:1] Reserved<br>b[0] '1' = Enable, '0' = Clear |
| **0x00CC** | FIFO0_RW_PORT | b [15:0] Read/Write Port.<br>(Word access only) |
| **FIFO Channel 1** | | |
| **0x00D0** | FIFO1_ID | b[15:0] ID Register = 0x2011 |

**Table 4: DM7820/DM9820 Memory Map**

| Offset (Hex) | Register Name | Register Function |
|---|---|---|
| **0x00D2** | FIFO1_INT | b [15:8] Interrupt Status – '1' = Interrupt condition has occurred. Write '1' to clear.<br>b[7:0] Interrupt Enable – '1' = Interrupt is enabled, '0' = disabled<br>Interrupt source are:<br>7 Reserved<br>6 Reserved<br>5 Underflow<br>4 Overflow<br>3 Empty<br>2 Full<br>1 Write Request<br>0 Read Request |
| **0x00D4** | FIFO1_IN_CLK | b[15:5] Reserved<br>b[4:0] Input Clock Select<br>31 = PCI Write<br>30 = PCI Read<br>29-16 = Interrupts[13-0]<br>15-0 = Clock_Bus [15-0] |
| **0x00D6** | FIFO1_OUT_CLK | b[15:5] Reserved<br>b[4:0] Input Clock Select<br>31 = PCI Write<br>30 = PCI Read<br>29-16 = Interrupts[13-0]<br>15-0 = Clock_Bus [15-0] |
| **0x00D8** | FIFO1_IN_DATA_DREQ | b[15:10]Reserved<br>b[9:8] DREQ1 Source<br>3 = Not Full<br>2 = Write Request<br>1 = Not Empty<br>0 = Read Request<br>b[7:4] Reserved<br>b[3:0] Input Data Select<br>3 = Incremental Encoder B1<br>2 = Incremental Encoder B0<br>1 = Port 1<br>0 = PCI Data |
| **0x00DA** | FIFO1_CON_STAT | b[15:10]Reserved<br>b[9] Write Request (non-sticky)<br>b[8] Read Request (non-sticky)<br>b[7:1] Reserved<br>b[0] '1' = Enable, '0' = Clear |
| **0x00DC** | FIFO1_RW_PORT | b [15:0] Read/Write Port.<br>(Word access only) |
| **Programmable Clock 0** | | |
| **0x0100** | PRGCLK0_ID | b[15:0] ID Register = 0x1000 |

**Table 4: DM7820/DM9820 Memory Map**

| Offset (Hex) | Register Name | Register Function |
|---|---|---|
| **0x0102** | PRGCLK0_MODE | b[15:2] Reserved<br>b[1:0] '00' = Disabled<br>'01' = Continuous<br>'10' = Reserved<br>'11' = One-Shot |
| **0x0104** | PRGCLK0_CLK | b[15:4] Reserved<br>b[3:0] Master Clock Source<br>15-0 = Clock_Bus [15-0] |
| **0x0106** | PRGCLK0_START_STOP | b[15:13]Reserved<br>b[12:8] Stop Clock<br>31-16 = Interrupt_Bus[15-0]<br>15-1 = Clock_Bus [15-1]<br>0 = No Stop Clock<br>b[7:5] Reserved<br>b[4:0] Start Trigger<br>31-16 = Interrupt_Bus[15-0]<br>15-1 = Clock_Bus [15-1]<br>0 = Start Immediate |
| **0x0108** | PRGCLK0_PERIOD | b[15:0] Period of Clock<br>Output frequency is:<br>$\frac{Master\_Clock\_Frequency}{(PRG\_CLK\_PERIOD+1)}$ |
| **Programmable Clock 1** | | |
| **0x0140** | PRGCLK1_ID | b[15:0] ID Register = 0x1000 |
| **0x0142** | PRGCLK1_MODE | b[15:2] Reserved<br>b[1:0] '00' = Disabled<br>'01' = Continuous<br>'10' = Reserved<br>'11' = One-Shot |
| **0x0144** | PRGCLK1_CLK | b[15:4] Reserved<br>b[3:0] Master Clock Source<br>15-0 = Clock_Bus [15-0] |
| **0x0146** | PRGCLK1_START_STOP | b[15:13]Reserved<br>b[12:8] Stop Clock<br>31-16 = Interrupt_Bus[15-0]<br>15-1 = Clock_Bus [15-1]<br>0 = No Stop Clock<br>b[7:5] Reserved<br>b[4:0] Start Trigger<br>31-16 = Interrupt_Bus[15-0]<br>15-1 = Clock_Bus [15-1]<br>0 = Start Immediate |
| **0x0148** | PRGCLK1_PERIOD | b[15:0] Period of Clock<br>Output frequency is:<br>$\frac{Master\_Clock\_Frequency}{(PRG\_CLK\_PERIOD+1)}$ |
| **Programmable Clock 2** | | |
| **0x0180** | PRGCLK2_ID | b[15:0] ID Register = 0x1000 |

**Table 4: DM7820/DM9820 Memory Map**

| Offset (Hex) | Register Name | Register Function |
|---|---|---|
| **0x0182** | PRGCLK2_MODE | b[15:2] Reserved<br>b[1:0] '00' = Disabled<br>'01' = Continuous<br>'10' = Reserved<br>'11' = One-Shot |
| **0x0184** | PRGCLK2_CLK | b[15:4] Reserved<br>b[3:0] Master Clock Source<br>15-0 = Clock_Bus [15-0] |
| **0x0186** | PRGCLK2_START_STOP | b[15:13]Reserved<br>b[12:8] Stop Clock<br>31-16 = Interrupt_Bus[15-0]<br>15-1 = Clock_Bus [15-1]<br>0 = No Stop Clock<br>b[7:5] Reserved<br>b[4:0] Start Trigger<br>31-16 = Interrupt_Bus[15-0]<br>15-1 = Clock_Bus [15-1]<br>0 = Start Immediate |
| **0x0188** | PRGCLK2_PERIOD | b[15:0] Period of Clock<br>Output frequency is:<br>$\frac{Master\_Clock\_Frequency}{(PRG\_CLK\_PERIOD+1)}$ |
| **Programmable Clock 3** | | |
| **0x01C0** | PRGCLK3_ID | b[15:0] ID Register = 0x1000 |
| **0x01C2** | PRGCLK3_MODE | b[15:2] Reserved<br>b[1:0] '00' = Disabled<br>'01' = Continuous<br>'10' = Reserved<br>'11' = One-Shot |
| **0x01C4** | PRGCLK3_SOURCE | b[15:4] Reserved<br>b[3:0] Master Clock Source<br>15-0 = Clock_Bus [15-0] |
| **0x01C6** | PRGCLK3_START_STOP | b[15:13]Reserved<br>b[12:8] Stop Clock<br>31-16 = Interrupt_Bus[15-0]<br>15-1 = Clock_Bus [15-1]<br>0 = No Stop Clock<br>b[7:5] Reserved<br>b[4:0] Start Trigger<br>31-16 = Interrupt_Bus[15-0]<br>15-1 = Clock_Bus [15-1]<br>0 = Start Immediate |
| **0x01C8** | PRGCLK3_PERIOD | b[15:0] Period of Clock<br>Output frequency is:<br>$\frac{Master\_Clock\_Frequency}{(PRG\_CLK\_PERIOD+1)}$ |
| **Advanced Interrupt 0** | | |
| **0x0200** | ADVINT0_ID | b[15:0] ID Register = 0x0001 |

**Table 4: DM7820/DM9820 Memory Map**

| Offset (Hex) | Register Name | Register Function |
|---|---|---|
| **0x0202** | ADVINT0_INT_MODE | b[1:0] Interrupt Mode<br>3 = Event Mode<br>2 = Match Mode<br>1 = Strobe Mode<br>0 = Disabled |
| **0x0204** | ADVINT0_CLK | b[3:0] Sample Clock Source<br>15-0 = Clock_Bus [15-0] |
| **0x0206** | Reserved | |
| **0x0208** | ADVINT0_PORT0_MASK | b[15:0] Port 0 Mask<br>'0' = Bit is used for match/event<br>'1' = Bit is ignored |
| **0x020A** | ADVINT0_PORT1_MASK | b[15:0] Port 1 Mask<br>'0' = Bit is used for match/event<br>'1' = Bit is ignored |
| **0x020C** | ADVINT0_PORT2_MASK | b[15:0] Port 2 Mask<br>'0' = Bit is used for match/event<br>'1' = Bit is ignored |
| **0x020E** | Reserved | |
| **0x0210** | ADVINT0_PORT0_CMP | b [15:0] Port 0 Compare – Value used for interrupt on match. |
| **0x0212** | ADVINT0_PORT1_CMP | b [15:0] Port 1 Compare – Value used for interrupt on match. |
| **0x0214** | ADVINT0_PORT2_CMP | b [15:0] Port 2 Compare – Value used for interrupt on match. |
| **0x0216** | Reserved | |
| **0x0218** | ADVINT0_PORT0_CAPT | b [15:0] Port 0 Capture – Value on Port 0 is written to this register when an interrupt occurs. |
| **0x021A** | ADVINT0_PORT1_CAPT | b [15:0] Port 1 Capture – Value on Port 1 is written to this register when an interrupt occurs. |
| **0x021C** | ADVINT0_PORT2_CAPT | b [15:0] Port 2 Capture – Value on Port 2 is written to this register when an interrupt occurs. |
| **0x021E** | Reserved | |
| **Advanced Interrupt 1** | | |
| **0x0240** | ADVINT1_ID | b[15:0] ID Register = 0x0001 |
| **0x0242** | ADVINT1_INT_MODE | b[1:0] Interrupt Mode<br>3 = Event Mode<br>2 = Match Mode<br>1 = Strobe Mode<br>0 = Disabled |
| **0x0244** | ADVINT1_CLK | b[3:0] Sample Clock Source<br>15-0 = Clock_Bus [15-0] |
| **0x0246** | Reserved | |
| **0x0248** | ADVINT1_PORT0_MASK | b[15:0] Port 0 Mask<br>'0' = Bit is used for match/event<br>'1' = Bit is ignored |
| **0x024A** | ADVINT1_PORT1_MASK | b[15:0] Port 1 Mask<br>'0' = Bit is used for match/event<br>'1' = Bit is ignored |

**Table 4: DM7820/DM9820 Memory Map**

| Offset (Hex) | Register Name | Register Function |
|---|---|---|
| **0x024C** | ADVINT1_PORT2_MASK | b[15:0] Port 2 Mask<br>'0' = Bit is used for match/event<br>'1' = Bit is ignored |
| **0x024E** | Reserved | |
| **0x0250** | ADVINT1_PORT0_CMP | b [15:0] Port 0 Compare – Value used for interrupt on match. |
| **0x0252** | ADVINT1_PORT1_CMP | b [15:0] Port 1 Compare – Value used for interrupt on match. |
| **0x0254** | ADVINT1_PORT2_CMP | b [15:0] Port 2 Compare – Value used for interrupt on match. |
| **0x0256** | Reserved | |
| **0x0258** | ADVINT1_PORT0_CAPT | b [15:0] Port 0 Capture – Value on Port 0 is written to this register when an interrupt occurs. |
| **0x025A** | ADVINT1_PORT1_CAPT | b [15:0] Port 1 Capture – Value on Port 1 is written to this register when an interrupt occurs. |
| **0x025C** | ADVINT1_PORT2_CAPT | b [15:0] Port 2 Capture – Value on Port 2 is written to this register when an interrupt occurs. |
| **0x025E** | Reserved | |
| **Dual Incremental Encoder 0** | | |
| **0x0280** | INCENC0_ID | b[15:0] ID Register = 0x0002 |
| **0x0282** | INCENC0_INT | b [11:8] Interrupt Status – '1' = Interrupt condition has occurred. Write '1' to clear.<br>b[3:0] Interrupt Enable – '1' = Interrupt is enabled, '0' = disabled<br>Interrupt source are:<br>3 Encoder B Negative Rollover<br>2 Encoder B Positive Rollover<br>1 Encoder A Negative Rollover<br>0 Encoder A Positive Rollover |
| **0x0284** | INCENC0_CLOCK | b[3:0] Master Clock Source<br>15-0 = Clock_Bus [15-0] |
| **0x0286** | INCENC0_MODE | b [15:8] Phase Filter – Writing a '1' to a specific bit masks out a phase transition.<br>b[7:6] Reserved<br>b[5] Differential Mode '1' = Pseudo differential mode, '0' = Single ended mode<br>b[4] Input Filter – '1' = Enable Input Filter, '0' = Disable Input Filter<br>b [3] Join – '1' = Operate as single 32-bit Encoder, '0' = Operate as two 16-bit Encoders.<br>b [2] '0' = External Index is disabled, '1' = External Index is enabled.<br>b [1] Hold Register – '1' = Hold values register, '0' = Allow value register to change.<br>b [0] Count Enable – '1' = Encoder is enabled, '0' = Encoder is cleared. |
| **0x0288** | INCENC0_VALUEA | b[15:0] Value for Encoder A |
| **0x028A** | INCENC0_VALUEB | b[15:0] Value for Encoder B |
| **Dual Incremental Encoder 1** | | |

**Table 4: DM7820/DM9820 Memory Map**

| Offset (Hex) | Register Name | Register Function |
|---|---|---|
| **0x02C0** | INCENC1_ID | b[15:0] ID Register = 0x0002 |
| **0x02C2** | INCENC1_INT | b [11:8] Interrupt Status – '1' = Interrupt condition has occurred. Write '1' to clear.<br>b[3:0] Interrupt Enable – '1' = Interrupt is enabled, '0' = disabled<br>Interrupt source are:<br>3 Encoder B Negative Rollover<br>2 Encoder B Positive Rollover<br>1 Encoder A Negative Rollover<br>0 Encoder A Positive Rollover |
| **0x02C4** | INCENC1_CLOCK | b[3:0] Master Clock Source<br>15-0 = Clock_Bus [15-0] |
| **0x02C6** | INCENC1_MODE | b [15:8] Phase Filter – Writing a '1' to a specific bit masks out a phase transition.<br>b[7:6] Reserved<br>b[5] Differential Mode '1' = Pseudo differential mode, '0' = Single ended mode<br>b[4] Input Filter – '1' = Enable Input Filter, '0' = Disable Input Filter<br>b [3] Join – '1' = Operate as single 32-bit Encoder, '0' = Operate as two 16-bit Encoders.<br>b [2] '0' = External Index is disabled, '1' = External Index is enabled.<br>b [1] Hold Register – '1' = Hold values register, '0' = Allow value register to change.<br>b[0] Count Enable – '1' = Encoder is enabled, '0' = Encoder is cleared. |
| **0x02C8** | INCENC1_VALUEA | b[15:0] Value for Encoder A |
| **0x02CA** | INCENC1_VALUEB | b[15:0] Value for Encoder B |
| **Pulse Width Modulator 0** | | |
| **0x0300** | PWM0_ID | b[15:0] ID Register = 0x0003 |
| **0x0302** | PWM0_MODE | b[0] '1' = Enable PWM, '0' = Disable PWM |
| **0x0304** | PWM0_CLK | b[7:4] Period Clock Source<br>15-0 = Clock_Bus [15-0]<br>b[3:0] Width Clock Source<br>15-0 = Clock_Bus [15-0] |
| **0x0306** | Reserved | |
| **0x0308** | PWM0_PERIOD | b[15:0] Period of PWM Cycle is:<br>$\frac{Width\_Clock\_Frequency}{(PWMx\_PERIOD+1)}$ |
| **0x030A-0x030E** | Reserved | |
| **0x0310** | PWM0_WIDTHA | b[15:0] Width of output A pulse in Period Clock cycles |
| **0x0312** | Reserved | |
| **0x0314** | PWM0_WIDTHB | b[15:0] Width of output B pulse in Period Clock cycles |
| **0x0316** | Reserved | |
| **0x0318** | PWM0_WIDTHC | b[15:0] Width of output C pulse in Period Clock cycles |
| **0x031A** | Reserved | |

**Table 4: DM7820/DM9820 Memory Map**

| Offset (Hex) | Register Name | Register Function |
|---|---|---|
| **0x031C** | PWM0_WIDTHD | b[15:0] Width of output D pulse in Period Clock cycles |
| **0x031E** | Reserved | |
| **Pulse Width Modulator 1** | | |
| **0x0340** | PWM1_ID | b[15:0] ID Register = 0x0003 |
| **0x0342** | PWM1_MODE | b[0] '1' = Enable PWM, '0' = Disable PWM |
| **0x0344** | PWM1_CLK | b[7:4] Period Clock Source<br>15-0 = Clock_Bus [15-0]<br>b[3:0] Width Clock Source<br>15-0 = Clock_Bus [15-0] |
| **0x0346** | Reserved | |
| **0x0348** | PWM1_PERIOD | b[15:0] Period of PWM Cycle is:<br>$\frac{Width\_Clock\_Frequency}{(PWMx\_PERIOD+1)}$ |
| **0x034A-0x034E** | Reserved | |
| **0x0350** | PWM1_WIDTHA | b[15:0] Width of output A pulse in Period Clock cycles |
| **0x0352** | Reserved | |
| **0x0354** | PWM1_WIDTHB | b[15:0] Width of output B pulse in Period Clock cycles |
| **0x0356** | Reserved | |
| **0x0358** | PWM1_WIDTHC | b[15:0] Width of output C pulse in Period Clock cycles |
| **0x035A** | Reserved | |
| **0x035C** | PWM1_WIDTHD | b[15:0] Width of output D pulse in Period Clock cycles |
| **0x035E** | Reserved | |
| **82C54 Timer Counter A** | | |
| **0x1000** | TCA_COUNTER_0 | b[7:0] Counter 0 Register |
| **0x1004** | TCA_COUNTER_1 | b[7:0] Counter 1 Register |
| **0x1008** | TCA_COUNTER_2 | b[7:0] Counter 2 Register |
| **0x100C** | TCA_CON_WORD | b[7:0] Control Word Register |
| **82C54 Timer Counter B** | | |
| **0x1010** | TCB_COUNTER_0 | b[7:0] Counter 0 Register |
| **0x1014** | TCB_COUNTER_1 | b[7:0] Counter 1 Register |
| **0x1018** | TCB_COUNTER_2 | b[7:0] Counter 2 Register |
| **0x101C** | TCB_CON_WORD | b[7:0] Control Word Register |

# *Detailed Register Description*

The following sections provide a detailed description of the individual registers. In the following register description sections, each register is described by a register table. The first row of the table lists the bits, D15 through D0. The second row lists the field name for each bit. The third row lists the properties of that bit; 'R' = bit can be read, 'W' = bit can be written to, and 'C' = bit can be cleared. The last row lists the value of the bit after reset. The register table is then followed by a description of each of the fields where applicable. An "N/A" for the reset value indicates that the reset value is not applicable - read the field descriptions for more information.

Bits marked as "Reserved" in the field name are unused, and reads will always return their reset value. These bits should not be modified during writes for future compatibility.

# System Block

## FPGA_VERSION

This register provides the version and type ID of the Digital I/O FPGA. The version can be used to identify the specific build of the board. The type ID can be used to identify a particular feature set.

| 15 8 | 7 0 |
|---|---|
| TYPE_ID | VERSION |
| R,+xxxx xxxx | R, +xxxx xxxx |

| Field | Description |
|---|---|
| TYPE_ID | FPGA Type Identifier.<br>0x10 = Standard FPGA |
| VERSION | FPGA Version Identifier |

## SVN_VERSION

This register provides the source code revision control version. It is updated every time the FPGA is compiled.

| 15 0 |
|---|
| VERSION |
| R,+xxxx xxxx xxxx xxxx |

| Field | Description |
|---|---|
| VERSION | FPGA Source Version Identifier |

## BOARD_RESET

Writing a value of 0xA5A5 to this register resets the board. All internal registers are set to their default values.

**Note:** The 82C54 Timer/Counters are not affected by this register

| 15 0 |
|---|
| RESET |
| W,+0000 0000 0000 0000 |

| Field | Description |
|---|---|
| RESET | Write 0xA5A5 to reset the board. All other writes are |

| | |
|---|---|
| | ignored. Reads will return all zeros. |

## BRD_STAT

This register contains status information for the board.

| 15 | 1 | 0 |
|---|---|---|
| Reserved | | MSTR |
| R,+0 | | R,+N |

| Field | Description |
|---|---|
| MSTR | Indicates if the board is PCI master capable based on the rotary switch and jumper settings.<br>'0' = PCI Master<br>'1' = Not PCI Master |

## INT_ENABLE

This register controls which interrupt sources are used to generate a local interrupt.

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| FIFO1 | FIFO0 | PClk3 | PClk2 | PClk1 | PClk0 | PWM1 | PWM0 |
| RW,+0 | RW,+0 | RW,+0 | RW,+0 | RW,+0 | RW,+0 | RW,+0 | RW,+0 |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| Reserved | | IncEnc1 | IncEnc0 | Rsvd | 82C54 | AdvInt1 | AdvInt0 |
| R,+00 | | RW,+0 | RW,+0 | R,+0 | RW,+0 | RW,+0 | RW,+0 |

| Field | Description |
|---|---|
| AdvInt0 | Interrupt from Advance Interrupt block at 0x0200<br>'0' = Interrupt Disabled<br>'1' = Interrupt Enabled |
| AdvInt1 | Interrupt from Advance Interrupt block at 0x0240<br>'0' = Interrupt Disabled<br>'1' = Interrupt Enabled |
| 82C54 | Interrupt 82C54 Timer/Counter block at 0x0080<br>'0' = Interrupt Disabled<br>'1' = Interrupt Enabled |
| IncEnc0 | Interrupt from Incremental Encoder block at 0x0280<br>'0' = Interrupt Disabled<br>'1' = Interrupt Enabled |
| IncEnc1 | Interrupt from Incremental Encoder block at 0x02C0<br>'0' = Interrupt Disabled<br>'1' = Interrupt Enabled |
| PWM0 | Interrupt from Pulse Width Modulator block at 0x0300<br>'0' = Interrupt Disabled<br>'1' = Interrupt Enabled |

| | |
|---|---|
| PWM1 | Interrupt from Pulse Width Modulator block at 0x0340<br>'0' = Interrupt Disabled<br>'1' = Interrupt Enabled |
| PClk0 | Interrupt from Programmable Clock block at 0x0100<br>'0' = Interrupt Disabled<br>'1' = Interrupt Enabled |
| PClk1 | Interrupt from Programmable Clock block at 0x0140<br>'0' = Interrupt Disabled<br>'1' = Interrupt Enabled |
| PClk2 | Interrupt from Programmable Clock block at 0x0180<br>'0' = Interrupt Disabled<br>'1' = Interrupt Enabled |
| PClk3 | Interrupt from Programmable Clock block at 0x01C0<br>'0' = Interrupt Disabled<br>'1' = Interrupt Enabled |
| FIFO0 | Interrupt from FIFO block at 0x00C0<br>'0' = Interrupt Disabled<br>'1' = Interrupt Enabled |
| FIFO1 | Interrupt from FIFO block at 0x00D0<br>'0' = Interrupt Disabled<br>'1' = Interrupt Enabled |

## INT_STATUS

This register shows if any of the interrupt conditions has occurred. This is a sticky register – bits remain set until cleared by writing a '1'. Interrupts do not have to be enabled in INT_ENABLE in order for status bits to be set.

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| FIFO1 | FIFO0 | PClk3 | PClk2 | PClk1 | PClk0 | PWM1 | PWM0 |
| RC,+0 | RC,+0 | RC,+0 | RC,+0 | RC,+0 | RC,+0 | RC,+0 | RC,+0 |

| 7 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|
| Reserved | IncEnc1 | IncEnc0 | Rsvd | 82C54 | AdvInt1 | AdvInt0 |
| R,+00 | RC,+0 | RC,+0 | R,+0 | RC,+0 | RC,+0 | RC,+0 |

| Field | Description |
|---|---|
| AdvInt0 | Interrupt from Advance Interrupt block at 0x0200<br>'0' = Interrupt has not occurred<br>'1' = Interrupt has occurred.<br>Write '1' to clear. |
| AdvInt1 | Interrupt from Advance Interrupt block at 0x0240<br>'0' = Interrupt has not occurred<br>'1' = Interrupt has occurred.<br>Write '1' to clear. |
| 82C54 | Interrupt 82C54 Timer/Counter block at 0x0080<br>'0' = Interrupt has not occurred<br>'1' = Interrupt has occurred.<br>Write '1' to clear. |
| IncEnc0 | Interrupt from Incremental Encoder block at 0x0280<br>'0' = Interrupt has not occurred |

| | |
|---|---|
| | ’1’ = Interrupt has occurred.<br>Write ‘1’ to clear. |
| IncEnc1 | Interrupt from Incremental Encoder block at 0x02C0<br>’0’ = Interrupt has not occurred<br>’1’ = Interrupt has occurred.<br>Write ‘1’ to clear. |
| PWM0 | Interrupt from Pulse Width Modulator block at 0x0300<br>’0’ = Interrupt has not occurred<br>’1’ = Interrupt has occurred.<br>Write ‘1’ to clear. |
| PWM1 | Interrupt from Pulse Width Modulator block at 0x0340<br>’0’ = Interrupt has not occurred<br>’1’ = Interrupt has occurred.<br>Write ‘1’ to clear. |
| PClk0 | Interrupt from Programmable Clock block at 0x0100<br>’0’ = Interrupt has not occurred<br>’1’ = Interrupt has occurred.<br>Write ‘1’ to clear. |
| PClk1 | Interrupt from Programmable Clock block at 0x0140<br>’0’ = Interrupt has not occurred<br>’1’ = Interrupt has occurred.<br>Write ‘1’ to clear. |
| PClk2 | Interrupt from Programmable Clock block at 0x0180<br>’0’ = Interrupt has not occurred<br>’1’ = Interrupt has occurred.<br>Write ‘1’ to clear. |
| PClk3 | Interrupt from Programmable Clock block at 0x01C0<br>’0’ = Interrupt has not occurred<br>’1’ = Interrupt has occurred.<br>Write ‘1’ to clear. |
| FIFO0 | Interrupt from FIFO block at 0x00C0<br>’0’ = Interrupt has not occurred<br>’1’ = Interrupt has occurred.<br>Write ‘1’ to clear. |
| FIFO1 | Interrupt from FIFO block at 0x00D0<br>’0’ = Interrupt has not occurred<br>’1’ = Interrupt has occurred.<br>Write ‘1’ to clear. |

## Standard I/O

A diagram of the standard I/O is shown in Figure 4. Each digital I/O pin can be an input, output, or peripheral output. The peripheral outputs are the Pulse Width Modulators, FIFO, Timer/Counters, etc.

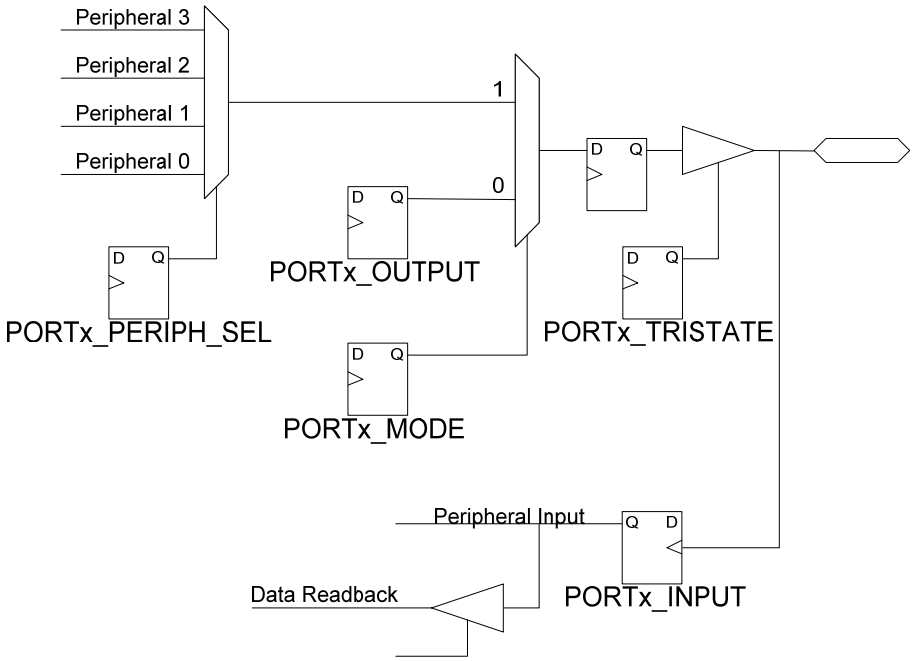



**Figure 4: Digital I/O Block Diagram**

## PORTx_OUTPUT

Sets the value for Port 0, Port 1, or Port 2 when it is a standard output.

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| Px_15 | Px_14 | Px_13 | Px_12 | Px_11 | Px_10 | Px_9 | Px_8 |
| RW,+0 | RW,+0 | RW,+0 | RW,+0 | RW,+0 | RW,+0 | RW,+0 | RW,+0 |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| Px_7 | Px_6 | Px_5 | Px_4 | Px_3 | Px_2 | Px_1 | Px_0 |
| RW,+0 | RW,+0 | RW,+0 | RW,+0 | RW,+0 | RW,+0 | RW,+0 | RW,+0 |

| Field | Description |
|---|---|
| Px_[15:0] | Value to output<br>'0' = Low<br>'1' = High |

## PORTx_INPUT

Returns the current value of Port 0, Port 1, or Port 2.

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| Px_15 | Px_14 | Px_13 | Px_12 | Px_11 | Px_10 | Px_9 | Px_8 |
| R,+0 | R,+0 | R,+0 | R,+0 | R,+0 | R,+0 | R,+0 | R,+0 |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| Px_7 | Px_6 | Px_5 | Px_4 | Px_3 | Px_2 | Px_1 | Px_0 |
| R,+0 | R,+0 | R,+0 | R,+0 | R,+0 | R,+0 | R,+0 | R,+0 |

| Field | Description |
|---|---|
| Px_[15:0] | Current pin value<br>'0' = Pin is Low<br>'1' = Pin is High |

## PORTx_TRISTATE

This register selects if each bit in Port 0, Port 1, or Port 2 is an input or an output.

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| Px_15 | Px_14 | Px_13 | Px_12 | Px_11 | Px_10 | Px_9 | Px_8 |
| RW,+0 | RW,+0 | RW,+0 | RW,+0 | RW,+0 | RW,+0 | RW,+0 | RW,+0 |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| Px_7 | Px_6 | Px_5 | Px_4 | Px_3 | Px_2 | Px_1 | Px_0 |
| RW,+0 | RW,+0 | RW,+0 | RW,+0 | RW,+0 | RW,+0 | RW,+0 | RW,+0 |

| Field | Description |
|---|---|
| Px_[15:0] | Select input or output<br>'0' = Input<br>'1' = Output |

## PORTx_MODE

Selects if each pin in Port 0, Port 1, or Port 2 is a standard I/O (controlled by PORTx_TRISTATE) or a peripheral output (controlled by PORTx_PERIPH_SEL).

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| Px_15 | Px_14 | Px_13 | Px_12 | Px_11 | Px_10 | Px_9 | Px_8 |
| RW,+0 | RW,+0 | RW,+0 | RW,+0 | RW,+0 | RW,+0 | RW,+0 | RW,+0 |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| Px_7 | Px_6 | Px_5 | Px_4 | Px_3 | Px_2 | Px_1 | Px_0 |
| RW,+0 | RW,+0 | RW,+0 | RW,+0 | RW,+0 | RW,+0 | RW,+0 | RW,+0 |

| Field | Description |
|---|---|
| Px_[15:0] | Port Mode<br>'0' = Standard I/O (controlled by PORTx_ TRISTATE)<br>'1' = Peripheral (controlled by PORTx_PERIPH_SEL) |

## PORTx_PERIPH_SEL_L

This register selects the peripheral for Port 0, Port 1, or Port 2 when it is a peripheral output (i.e. PORTx_MODE[] = '1'). This register selects the peripheral for bits [7:0].

| 15 14 | 13 12 | 11 10 | 9 8 |
|---|---|---|---|
| Px_7 | Px_6 | Px_5 | Px_4 |
| RW,+00 | RW,+00 | RW,+00 | RW,+00 |

| 7 6 | 5 4 | 3 2 | 1 0 |
|---|---|---|---|
| Px_3 | Px_2 | Px_1 | Px_0 |
| RW,+00 | RW,+00 | RW,+00 | RW,+00 |

## PORTx_PERIPH_SEL_H

This register selects the peripheral for Port 0, Port 1, or Port 2 when it is a peripheral output (i.e. PORTx_MODE[] = '1'). This register selects the peripheral for bits [15:8].

| 15 14 | 13 12 | 11 10 | 9 8 |
|---|---|---|---|
| Px_15 | Px_14 | Px_13 | Px_12 |
| RW,+00 | RW,+00 | RW,+00 | RW,+00 |

| 7 6 | 5 4 | 3 2 | 1 0 |
|---|---|---|---|
| Px_11 | Px_10 | Px_9 | Px_8 |
| RW,+00 | RW,+00 | RW,+00 | RW,+00 |

**Table 5: Peripheral Outputs**

| Pin | PORTx_PERIPH_SEL | | | |
|---|---|---|---|---|
| | 00 | 01 | 10 | 11 |
| Port0[0] | | | FIFO0_Out[0] | FIFO1_Out[0] |
| Port0[1] | | | FIFO0_Out[1] | FIFO1_Out[1] |
| Port0[2] | | | FIFO0_Out[2] | FIFO1_Out[2] |
| Port0[3] | | | FIFO0_Out[3] | FIFO1_Out[3] |
| Port0[4] | | | FIFO0_Out[4] | FIFO1_Out[4] |
| Port0[5] | | | FIFO0_Out[5] | FIFO1_Out[5] |
| Port0[6] | | | FIFO0_Out[6] | FIFO1_Out[6] |
| Port0[7] | | | FIFO0_Out[7] | FIFO1_Out[7] |
| Port0[8] | | | FIFO0_Out[8] | FIFO1_Out[8] |
| Port0[9] | | | FIFO0_Out[9] | FIFO1_Out[9] |
| Port0[10] | | | FIFO0_Out[10] | FIFO1_Out[10] |
| Port0[11] | | | FIFO0_Out[11] | FIFO1_Out[11] |
| Port0[12] | | | FIFO0_Out[12] | FIFO1_Out[12] |
| Port0[13] | | | FIFO0_Out[13] | FIFO1_Out[13] |
| Port0[14] | | | FIFO0_Out[14] | FIFO1_Out[14] |
| Port0[15] | | | FIFO0_Out[15] | FIFO1_Out[15] |
| Port1[0] | | | FIFO0_Out[0] | FIFO1_Out[0] |
| Port1[1] | | | FIFO0_Out[1] | FIFO1_Out[1] |
| Port1[2] | | | FIFO0_Out[2] | FIFO1_Out[2] |
| Port1[3] | | | FIFO0_Out[3] | FIFO1_Out[3] |

**Table 5: Peripheral Outputs**

| Pin | PORTx_PERIPH_SEL | | | |
|---|---|---|---|---|
| | 00 | 01 | 10 | 11 |
| Port1[4] | | | FIFO0_Out[4] | FIFO1_Out[4] |
| Port1[5] | | | FIFO0_Out[5] | FIFO1_Out[5] |
| Port1[6] | | | FIFO0_Out[6] | FIFO1_Out[6] |
| Port1[7] | | | FIFO0_Out[7] | FIFO1_Out[7] |
| Port1[8] | | | FIFO0_Out[8] | FIFO1_Out[8] |
| Port1[9] | | | FIFO0_Out[9] | FIFO1_Out[9] |
| Port1[10] | | | FIFO0_Out[10] | FIFO1_Out[10] |
| Port1[11] | | | FIFO0_Out[11] | FIFO1_Out[11] |
| Port1[12] | | | FIFO0_Out[12] | FIFO1_Out[12] |
| Port1[13] | | | FIFO0_Out[13] | FIFO1_Out[13] |
| Port1[14] | | | FIFO0_Out[14] | FIFO1_Out[14] |
| Port1[15] | | | FIFO0_Out[15] | FIFO1_Out[15] |
| Port2[0] | PWM0_A+ | | FIFO0_Out[0] | FIFO1_Out[0] |
| Port2[1] | PWM0_A- | | FIFO0_Out[1] | FIFO1_Out[1] |
| Port2[2] | PWM0_B+ | TC_A0_OUT | FIFO0_Out[2] | FIFO1_Out[2] |
| Port2[3] | PWM0_B- | TC_A1_OUT | FIFO0_Out[3] | FIFO1_Out[3] |
| Port2[4] | PWM0_C+ | TC_A2_OUT | FIFO0_Out[4] | FIFO1_Out[4] |
| Port2[5] | PWM0_C- | TC_B0_OUT | FIFO0_Out[5] | FIFO1_Out[5] |
| Port2[6] | PWM0_D+ | TC_B1_OUT | FIFO0_Out[6] | FIFO1_Out[6] |
| Port2[7] | PWM0_D- | TC_B2_OUT | FIFO0_Out[7] | FIFO1_Out[7] |
| Port2[8] | PWM1_A+ | ProgClk0_OUT | FIFO0_Out[8] | FIFO1_Out[8] |
| Port2[9] | PWM1_A- | ProgClk1_OUT | FIFO0_Out[9] | FIFO1_Out[9] |
| Port2[10] | PWM1_B+ | ProgClk2_OUT | FIFO0_Out[10] | FIFO1_Out[10] |
| Port2[11] | PWM1_B- | ProgClk3_OUT | FIFO0_Out[11] | FIFO1_Out[11] |
| Port2[12] | PWM1_C+ | Strobe1_pos | FIFO0_Out[12] | FIFO1_Out[12] |
| Port2[13] | PWM1_C- | Strobe2_pos | FIFO0_Out[13] | FIFO1_Out[13] |
| Port2[14] | PWM1_D+ | Strobe1_neg | FIFO0_Out[14] | FIFO1_Out[14] |
| Port2[15] | PWM1_D- | Strobe2_neg | FIFO0_Out[15] | FIFO1_Out[15] |

## STROBE_STATUS

This register can be used to check the status of the strobe signals, as well as configure the strobes as outputs.

| 15 … 10 | 9 | 8 |
|---|---|---|
| Reserved | STR2_TRI | STR1_TRI |
| R,+0000 00 | RW,+0 | RW,+0 |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| Reserved | | STR2_OUT | STR1_OUT | Reserved | | STR2_IN | STR1_IN |
| R,+00 | | RW,+0 | RW,+0 | R,+00 | | R,+x | R,+x |

| Field | Description |
|---|---|
| STR1_IN | Current State of Strobe 1<br>'0' = Low<br>'1' = High |

| Field | Description |
|---|---|
| STR2_IN | Current State of Strobe 2<br>'0' = Low<br>'1' = High |
| STR1_OUT | Value to drive on Strobe 1 when an output<br>'0' = Low<br>'1' = High |
| STR2_OUT | Value to drive on Strobe 2 when an output<br>'0' = Low<br>'1' = High |
| STR1_TRI | Selects Input or Output for strobe 1<br>'0' = Input<br>'1' = Output |
| STR2_TRI | Selects Input or Output for strobe 2<br>'0' = Input<br>'1' = Output |

## 82C54 Timer Counter Control

The Timer/Counter Control section is used to select the clock, gates and interrupt sources for the 82C54 Timer/Counters. The actual Timer/Counter registers are found in the 82C54 Timer Counter n section on page 55.

### TC_ID

ID register to identify the Timer/Counter Block.

| 15 | 0 |
|---|---|
| ID_Register | |
| R | |

| Field | Description |
|---|---|
| ID_Register15:0] | Value of 0x1001 indicates Timer Counter Control Block |

### TC_INT

Enable and status for the interrupts generated by the 82C54 Timer Counters

| 15 14 | 13 8 | 7 6 | 5 0 |
|---|---|---|---|
| Reserved | INT_STAT[5:0] | Reserved | INT_ENA[5:0] |
| R,+00 | RC,+0 | R,+00 | RW,+0 |

| Field | Description |
|---|---|
| INT_STAT[5:0] | Interrupt Status – '1' = Interrupt condition has occurred. Write '1' to clear. Interrupts are asserted on the positive edge of the clock. |

| Field | Description |
|---|---|
| INT_ENA[5:0] | Interrupt Enable – '1' = Interrupt is enabled, '0' = disabled<br>Interrupt source are:<br>5 TC B2<br>4 TC B1<br>3 TC B0<br>2 TC A2<br>1 TC A1<br>0 TC A0 |

## TC_xy_CONTROL

This register selects the input clock and gate source for the 82C54 Timer Counters. Note that the maximum input frequency to the Timer/Counters is 10 MHz. Also, no provision is made in hardware to prevent a Timer/Counter from using its own output clock as its input clock.

| 15 13 | 12 8 | 7 4 | 3 0 |
|---|---|---|---|
| Reserved | GATE_SEL[4:0] | Reserved | CLOCK_SEL[3:0] |
| R,+00 | RW,+0 | R,+00 | RW,+0 |

| Field | Description |
|---|---|
| GATE_SEL[4:0] | Selects the gate input to this channel of the Timer/ Counter. Value definitions are:<br><br>31 Port2[15]<br>…<br>16 Port2[0]<br>15 Inverted Strobe2<br>14 Inverted Strobe1<br>13 Strobe2<br>12 Strobe1<br>11 Prog. Clock 3<br>10 Prog. Clock 2<br>9 Prog. Clock 1<br>8 Prog. Clock 0<br>7 82C54 TC B2<br>6 82C54 TC B1<br>5 82C54 TC B0<br>4 82C54 TC A2<br>3 82C54 TC A1<br>2 82C54 TC A0<br>1 '1'<br>0 '0' |

| Field | Description |
|---|---|
| CLOCK_SEL[3:0] | Selects the clock input to this channel of the Timer/ Counter. Value definitions are:<br><br>15 Inverted Strobe2<br>14 Inverted Strobe1<br>13 Strobe2<br>12 Strobe1<br>11 Prog. Clock 3<br>10 Prog. Clock 2<br>9 Prog. Clock 1<br>8 Prog. Clock 0<br>7 82C54 TC B2<br>6 82C54 TC B1<br>5 82C54 TC B0<br>4 82C54 TC A2<br>3 82C54 TC A1<br>2 82C54 TC A0<br>1 Reserved<br>0 5 MHz |

## FIFO Channel n

The DM7820/DM9820 provides two FIFOs to buffer data going into and out of the board. Each FIFO is 4MB in size. The input strobe, output strobe, and data input for each FIFO can be individually selected. The output data is made available to the peripheral outputs, and also the PCI interface.

Each FIFO is attached to a DMA Channel in the PLX chip. FIFO0 is attached to DMA0, and FIFO1 is attached to DMA1.

### FIFOn_ID

ID register to identify a FIFO Block.

| 15 | 0 |
|---|---|
| ID_Register | |
| R | |

| Field | Description |
|---|---|
| ID_Register15:0] | Value of 0x2011 indicates SDRAM/FIFO Block |

### FIFOn_INT

Enable and status for the interrupts generated by the FIFOs. An Overflow condition occurs when the FIFO is full, and it is written to. It can also occur when the FIFO is written to too fast. An Underflow occurs when the FIFO is empty and the output clock toggles, or when the FIFO is read from too fast. When the FIFO is disabled, the “Full,” “Empty,” and both requests are asserted.

| 15 | 8 | 7 | 0 |
|---|---|---|---|
| INT_STAT[7:0] | | INT_ENA[7:0] | |

| RC,+0 | RW,+0 |
|---|---|

| Field | Description |
|---|---|
| INT_STAT[7:0] | Interrupt Status – '1' = Interrupt condition has occurred. Write '1' to clear. Interrupts are asserted on the positive edge of the clock. |
| INT_ENA[7:0] | Interrupt Enable – '1' = Interrupt is enabled, '0' = disabled<br>Interrupt source are:<br>7 Reserved<br>6 Reserved<br>5 Underflow<br>4 Overflow<br>3 Empty<br>2 Full<br>1 Write Request<br>0 Read Request |

## FIFOn_IN_CLK

This register selects the input clock to the FIFO. At every positive edge of the input clock, a word is read into the FIFO from the input source.

| 15 5 | 4 0 |
|---|---|
| Reserved | CLOCK_SEL[4:0] |
| R,+0 | RW,+0 |

| Field | Description |
|---|---|
| CLOCK_SEL[4:0] | Selects the input clock input to this FIFO channel. Value definitions are:<br><br>31 PCI Write to FIFOn_RW_PORT<br>30 PCI Read from FIFOn_RW_PORT<br>29 Prog. Clock 3 Interrupt<br>28 Prog. Clock 2 Interrupt<br>27 Prog. Clock 1 Interrupt<br>26 Prog. Clock 0 Interrupt<br>25 PWM1 Interrupt<br>24 PWM0 Interrupt<br>23 Reserved<br>22 Reserved<br>21 Incremental Encoder 1 Interrupt<br>20 Incremental Encoder 0 Interrupt<br>19 Reserved<br>18 82C54 Interrupt<br>17 Advanced Interrupt 1 Interrupt<br>16 Advanced Interrupt 0 Interrupt<br>15 Inverted Strobe2<br>14 Inverted Strobe1<br>13 Strobe2<br>12 Strobe1<br>11 Prog. Clock 3 |

| Field | Description |
|---|---|
| | 10 Prog. Clock 2<br>9 Prog. Clock 1<br>8 Prog. Clock 0<br>7 82C54 TC B2<br>6 82C54 TC B1<br>5 82C54 TC B0<br>4 82C54 TC A2<br>3 82C54 TC A1<br>2 82C54 TC A0<br>1 Reserved<br>0 25 MHz |

## FIFOn_OUT_CLK

This register selects the output clock to the FIFO. At every positive edge of the output clock, a new word available at the FIFO output.

| 15 5 | 4 0 |
|---|---|
| Reserved | CLOCK_SEL[4:0] |
| R,+0 | RW,+0 |

| Field | Description |
|---|---|
| CLOCK_SEL[4:0] | Selects the input clock input to this FIFO channel.<br>Value definitions are:<br><br>31 PCI Write to FIFOn_RW_PORT<br>30 PCI Read from FIFOn_RW_PORT<br>29 Prog. Clock 3 Interrupt<br>28 Prog. Clock 2 Interrupt<br>27 Prog. Clock 1 Interrupt<br>26 Prog. Clock 0 Interrupt<br>25 PWM1 Interrupt<br>24 PWM0 Interrupt<br>23 Reserved<br>22 Reserved<br>21 Incremental Encoder 1 Interrupt<br>20 Incremental Encoder 0 Interrupt<br>19 Reserved<br>18 82C54 Interrupt<br>17 Advanced Interrupt 1 Interrupt<br>16 Advanced Interrupt 0 Interrupt<br>15 Inverted Strobe2<br>14 Inverted Strobe1<br>13 Strobe2<br>12 Strobe1<br>11 Prog. Clock 3<br>10 Prog. Clock 2<br>9 Prog. Clock 1<br>8 Prog. Clock 0<br>7 82C54 TC B2<br>6 82C54 TC B1<br>5 82C54 TC B0 |

| Field | Description |
|---|---|
| | 4 82C54 TC A2<br>3 82C54 TC A1<br>2 82C54 TC A0<br>1 Reserved<br>0 25 MHz |

## FIFOn_IN_DATA_DREQ

This register selects the FIFO data input and PLX DMA Request source. For the "Write Request" and "Read Request" signals, internal buffers are monitored to signal when data can be sent into, and read from the FIFO. The "Write Request" is asserted when there is at least 256 words of space available in the FIFO, and negated when there is less than 128 words available. The "Read Request" is asserted when at least 256 words of data is in the FIFO, and negated when there is less than 128 words of data. Using these signals guarantees a burst of at least 128 words, which provides for efficient communication over the PCI bus, and robustly guards against over-run and under-run conditions. However, it does not allow for the FIFO to be completely filled of emptied.

The "Not Full" and "Not Empty" request source should only be used if the amount of data in the FIFO is known, or to finish filling/emptying the FIFO. The DMA engine on the PLX PCI9056 will complete an additional double-word transfer after the request is negated. Therefore, using the "Not Full" and "Not Empty" request source will generally result in an over-run/under-run condition whenever the signal is negated.

The DREQ signals are in an undefined state when the FIFO is disabled. The DMA engine should only be enabled after the FIFO is enabled (FIFOn_CON_STAT[ENA]).

| 15 10 | 9 8 | 7 2 | 1 0 |
|---|---|---|---|
| Reserved | DREQ_SRC[1:0] | Reserved | IN_DATA[1:0] |
| R,+0 | RW,+0 | R,+0 | RW,+0 |

| Field | Description |
|---|---|
| DREQ_SRC[1:0] | Selects the source for the DREQn signal to the PLX chip. Value definitions are:<br>3 = Not Full<br>2 = Write Request<br>1 = Not Empty<br>0 = Read Request |
| IN_DATA[1:0] | Selects the FIFO Input Data.<br>Value definitions for FIFO0 are:<br>3 = FIFO0 Output<br>2 = Port 2<br>1 = Port 0<br>0 = PCI Data<br>Value definitions for FIFO1 are:<br>3 = Incremental Encoder 1 Channel B Value<br>2 = Incremental Encoder 1 Channel A Value<br>1 = Port 1<br>0 = PCI Data |

### FIFOn_CON_STAT

This register is used to enable the FIFO. When the FIFO is disabled, it is internally reset, and all data is flushed from it.

This register also is used to read the current status of the “Write Request” and “Read Request” signals that are used for DMA Requests. For these signals, internal buffers are monitored to signal when data can be sent into, and read from the FIFO. The “Write Request” is asserted when there is at least 256 words of space available in the FIFO, and negated when there is less than 128 words available. The “Read Request” is asserted when at least 256 words of data is in the FIFO, and negated when there is less than 128 words of data. Using these signals guarantees a burst of at least 128 words, which provides for efficient communication over the PCI bus, and robustly guards against over-run and under-run conditions. However, it does not allow for the FIFO to be completely filled of emptied.

| 15 10 | 9 | 8 | 7 1 | 0 |
|---|---|---|---|---|
| Reserved | WRITE_REQ | READ_REQ | Reserved | ENA |
| R,+0 | R,+x | R,+0 | R,+0 | RW,+0 |

| Field | Description |
|---|---|
| WRITE_REQ | Current Write Request Status.<br>’0’ = Not ready to receive data<br>’1’ = Ready to receive data. |
| READ_REQ | Current Read Request Status.<br>’0’ = Not ready to send data<br>’1’ = Ready to send data. |
| ENA | FIFO Enable.<br>’0’ = FIFO is disabled and cleared<br>’1’ = FIFO is enabled. |

### FIFOn_RW_PORT

This register provides the PCI bus access to the FIFO. Reads from this register return the current data that is available at the output of the FIFO, and can be programmed to clock the next data out of the FIFO. Writes to this register can be programmed to write data into the FIFO.

Accesses to this register must be word (16-bit) or larger.

| 15 0 |
|---|
| DATA[15:0] |
| RW,+0 |

| Field | Description |
|---|---|
| DATA | The read or write data to the FIFO. |

## Programmable Clock n

There are four programmable clocks on the DM7820/DM9820. They can be cascaded. The Programmable Clocks use a master clock and divide it down by an integer,

An interrupt is generated at every positive edge of the clock output.

## PROGCLKn_ID

ID register to identify a Programmable Clock Block.

| 15 | 0 |
|---|---|
| ID_Register | |
| R | |

| Field | Description |
|---|---|
| ID_Register15:0] | Value of 0x1000 indicates Programmable Clock |

## PROGCLKn_MODE

Selects the mode that the Programmable Clock.

| 15 | 2 | 1 | 0 |
|---|---|---|---|
| Reserved | | MODE | |
| RW,+0 | | RW,+00 | |

| Field | Description |
|---|---|
| MODE | Selects continuous or one-shot mode. In continuous mode, the clock will generate a pulse train with the specified period. In one-shot mode, the clock will generate a single pulse one period time after it is started. The clock must be disabled when transitioning between modes.<br>'00' = Disabled<br>'01' = Continuous<br>'10' = Reserved<br>'11' = One-Shot – Must be disabled and re-enabled to produce a second pulse. |

## PRGCLKn_CLK

This register selects the master clock for the programmable clock. The clock should be disabled before modifying this register.

| 15 | 4 | 3 | 0 |
|---|---|---|---|
| Reserved | | CLOCK_SEL[3:0] | |
| R,+0 | | RW,+0 | |

| Field | Description |
|---|---|
| CLOCK_SEL[3:0] | Selects the master clock. Value definitions are:<br>15 Inverted Strobe2<br>14 Inverted Strobe1<br>13 Strobe2<br>12 Strobe1<br>11 Prog. Clock 3<br>10 Prog. Clock 2 |

| Field | Description |
|---|---|
| | 9 Prog. Clock 1<br>8 Prog. Clock 0<br>7 82C54 TC B2<br>6 82C54 TC B1<br>5 82C54 TC B0<br>4 82C54 TC A2<br>3 82C54 TC A1<br>2 82C54 TC A0<br>1 Reserved<br>0 25 MHz |

## PRGCLKn_START_STOP

This register selects the Start and Stop Trigger for the programmable clock. The clock will not begin generating an output until the first positive edge of the Start Trigger. The first edge of the programmable clock output will occur one period after the Start Trigger edge. If in continuous mode, the clock will continue to run until the first edge of the Stop Trigger. After the clock has stopped, it must be disabled and re-enabled for it to start again. The clock should be disabled before modifying this register.

| 15 13 | 12 8 | 7 5 | 4 0 |
|---|---|---|---|
| Reserved | STOP _TRG[4:0] | Reserved | START _TRG[4:0] |
| R,+0 | RW,+0 | R,+0 | RW,+0 |

| Field | Description |
|---|---|

| Field | Description |
|---|---|
| START_TRG[4:0] | Selects the start trigger. Value definitions are:<br>31 FIFO1 Interrupt<br>30 FIFO0 Interrupt<br>29 Prog. Clock 3 Interrupt<br>28 Prog. Clock 2 Interrupt<br>27 Prog. Clock 1 Interrupt<br>26 Prog. Clock 0 Interrupt<br>25 PWM1 Interrupt<br>24 PWM0 Interrupt<br>23 Reserved<br>22 Reserved<br>21 Incremental Encoder 1 Interrupt<br>20 Incremental Encoder 0 Interrupt<br>19 Reserved<br>18 82C54 Interrupt<br>17 Advanced Interrupt 1 Interrupt<br>16 Advanced Interrupt 0 Interrupt<br>15 Inverted Strobe2<br>14 Inverted Strobe1<br>13 Strobe2<br>12 Strobe1<br>11 Prog. Clock 3<br>10 Prog. Clock 2<br>9 Prog. Clock 1<br>8 Prog. Clock 0<br>7 82C54 TC B2<br>6 82C54 TC B1<br>5 82C54 TC B0<br>4 82C54 TC A2<br>3 82C54 TC A1<br>2 82C54 TC A0<br>1 Reserved<br>0 Start Immediate |

| Field | Description |
|---|---|
| STOP_TRG[4:0] | Selects the stop trigger. Value definitions are:<br>31 FIFO1 Interrupt<br>30 FIFO0 Interrupt<br>29 Prog. Clock 3 Interrupt<br>28 Prog. Clock 2 Interrupt<br>27 Prog. Clock 1 Interrupt<br>26 Prog. Clock 0 Interrupt<br>25 PWM1 Interrupt<br>24 PWM0 Interrupt<br>23 Reserved<br>22 Reserved<br>21 Incremental Encoder 1 Interrupt<br>20 Incremental Encoder 0 Interrupt<br>19 Reserved<br>18 82C54 Interrupt<br>17 Advanced Interrupt 1 Interrupt<br>16 Advanced Interrupt 0 Interrupt<br>15 Inverted Strobe2<br>14 Inverted Strobe1<br>13 Strobe2<br>12 Strobe1<br>11 Prog. Clock 3<br>10 Prog. Clock 2<br>9 Prog. Clock 1<br>8 Prog. Clock 0<br>7 82C54 TC B2<br>6 82C54 TC B1<br>5 82C54 TC B0<br>4 82C54 TC A2<br>3 82C54 TC A1<br>2 82C54 TC A0<br>1 Reserved<br>0 Do Not Stop |

## PROGCLKn_PERIOD

Sets the period of the programmable clock.

| 15 | 0 |
|---|---|
| PERIOD[15:0] | |
| RW,+0 | |

| Field | Description |
|---|---|
| PERIOD[15:0] | The frequency of the output clock is:<br>$\frac{Master\_Clock\_Frequency}{(PERIOD+1)}$ |

# Advanced Interrupt n

Two Advanced Interrupt block are provided that can generate an interrupt on a match, event, or strobe. The match and event interrupts are across all 48 digital I/O. The bits can be individually selected.

When an interrupt is generated, the data on all of the ports is latched into the Capture registers.

Bits are tested regardless of if a pin is an input or output.

A Match interrupt is generated when all un-masked bits in the Compare register match the input value of the port. This is when the following expression is true for ALL ports (x) and bits (y):

((PORTx[y] xor ADVINTn_PORTx_CMP[y]) and not ADVINTn_PORTx_MASK[y]) = '0'

An Event interrupt is generated when any un-masked input port bit changes. This is when the following expression is true for ANY ports (x) and bits (y). Note that the Capture register is updated at every interrupt or event:

((PORTx[y] xor ADVINTn_PORTx_CAPT[y]) and not ADVINTn_PORTx_MASK[y]) = '1'

## ADVINTn_ID

ID register to identify an Advanced Interrupt Block.

| 15 | 0 |
|---|---|
| ID_Register | |
| R | |

| Field | Description |
|---|---|
| ID_Register15:0] | Value of 0x0001 indicates Advanced Interrupt |

## ADVINTn_INT_MODE

Selects the mode for this interrupt. Event mode will generate an interrupt when any selected input pin changes. Match mode will generate an interrupt when the port(s) match a pre-set value (bits can be individually selected or masked). Strobe mode will generate an interrupt on the rising edge of the Strobe1 or Strobe2 signal.

| 15 2 | 1 0 |
|---|---|
| Reserved | MODE[1:0] |
| RW,+0 | RW,+0 |

| Field | Description |
|---|---|
| MODE[1:0] | Interrupt Mode. Value definitions are:<br>3 Event Mode<br>2 Match Mode<br>1 Strobe Mode<br>0 Disabled |

## ADVINTn_CLK

This register selects the clock source for sampling the ports when in Match or Compare mode. In Strobe mode, this register selects the actual strobe signal, and the 25 MHz clock always serves as the sampling clock.

| 15 4 | 3 0 |
|---|---|
| Reserved | CLOCK_SEL[3:0] |
| R,+0 | RW,+0 |

| Field | Description |
|---|---|
| CLOCK_SEL[3:0] | Selects the master clock. Value definitions are:<br>15 Inverted Strobe2<br>14 Inverted Strobe1<br>13 Strobe2<br>12 Strobe1<br>11 Prog. Clock 3<br>10 Prog. Clock 2<br>9 Prog. Clock 1<br>8 Prog. Clock 0<br>7 82C54 TC B2<br>6 82C54 TC B1<br>5 82C54 TC B0<br>4 82C54 TC A2<br>3 82C54 TC A1<br>2 82C54 TC A0<br>1 Reserved<br>0 25 MHz |

## ADVINTn_PORTx_MASK

This register determines if a bit is checked for the match and event interrupts.

**Note:** If Match mode is selected, and all bits are masked, an interrupt will be generated immediately.

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| Px_15 | Px_14 | Px_13 | Px_12 | Px_11 | Px_10 | Px_9 | Px_8 |
| RW,+0 | RW,+0 | RW,+0 | RW,+0 | RW,+0 | RW,+0 | RW,+0 | RW,+0 |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| Px_7 | Px_6 | Px_5 | Px_4 | Px_3 | Px_2 | Px_1 | Px_0 |
| RW,+0 | RW,+0 | RW,+0 | RW,+0 | RW,+0 | RW,+0 | RW,+0 | RW,+0 |

| Field | Description |
|---|---|
| Px_[15:0] | Bit mask. Bit definitions are:<br>'0' = Bit is used for match/event<br>'1' = Bit is ignored |

### ADVINTn_PORTx_CMP

The compare register is used for the Match interrupt. When all selected bits in this register match all selected bits on the input ports, an interrupt is generated.

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| Px_15 | Px_14 | Px_13 | Px_12 | Px_11 | Px_10 | Px_9 | Px_8 |
| RW,+0 | RW,+0 | RW,+0 | RW,+0 | RW,+0 | RW,+0 | RW,+0 | RW,+0 |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| Px_7 | Px_6 | Px_5 | Px_4 | Px_3 | Px_2 | Px_1 | Px_0 |
| RW,+0 | RW,+0 | RW,+0 | RW,+0 | RW,+0 | RW,+0 | RW,+0 | RW,+0 |

| Field | Description |
|---|---|
| Px_[15:0] | Compare Value. Bit definitions are:<br>'0' = Interrupt when this bit is '0' (when selected)<br>'1' = Interrupt when this bit is '1' (when selected) |

### ADVINTn_PORTx_CAPT

The Capture register latches the input ports when an interrupt is generated. All values are latched, regardless of the Mask register, or if the port is an input or output.

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|
| Px_15 | Px_14 | Px_13 | Px_12 | Px_11 | Px_10 | Px_9 | Px_8 |
| R,+0 | R,+0 | R,+0 | R,+0 | R,+0 | R,+0 | R,+0 | R,+0 |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| Px_7 | Px_6 | Px_5 | Px_4 | Px_3 | Px_2 | Px_1 | Px_0 |
| R,+0 | R,+0 | R,+0 | R,+0 | R,+0 | R,+0 | R,+0 | R,+0 |

| Field | Description |
|---|---|
| Px_[15:0] | Captured Value. Bit definitions are:<br>'0' = Input was '0' at last interrupt.<br>'1' = Input was '1' at last interrupt. |

## Dual Incremental Encoder n

Each Incremental Encoder block provides two encoder channels with 16 bit counters. These two channels can be linked into a single 32 counter.

An Incremental Encoder is used to detect the relative position of a shaft or linear actuator. A typical implementation is a slotted wheel with two optical sensors positioned such that when one sensor is positioned over a slot, the other is positioned between slots. The output of the optical sensors is shown in Figure 5, with one sensor named "A," and the other named "B." At every edge of the "A" or "B" input, the counter either increments or decrements. The direction can be interpreted from the state of the signals, i.e. which signal leads.

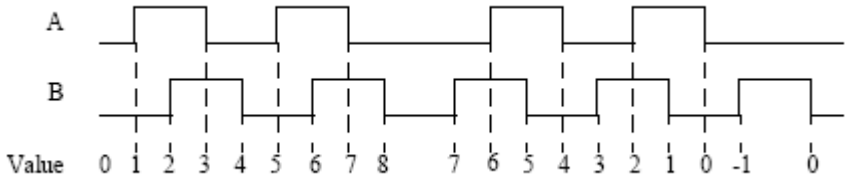



**Figure 5: Incremental Encoder Signals**

The encoders include a “Phase Filter” that prevents the counter from counting on certain transitions. This allows the encoders to count pulses, and other specialized applications.

Encoder inputs can be configured as single ended or pseudo-differential. In pseudo-differential mode, the “+” and “-“ inputs must be the inverse of each other in order for the encoder to see a change.

Digital filtering can be selected. With digital filtering, a transition on a line is only considered valid if it remains constant for four clock cycles. The clock can be selected.

Separate interrupts are generated for positive and negative rollover. Positive rollover occurs when the counter is at its maximum value, and receives a signal to count up. Negative rollover occurs when the counter is at 0, and receives a signal to count down. Because separate interrupts are generated, the counter can be easily expanded in software.

The Incremental Encoder inputs are show in Table 6 below.

**Table 6: Incremental Encoder Inputs**

| Pin | Encoder 0 | | Encoder 1 | |
|---|---|---|---|---|
| | Channel A | Channel B | Channel A | Channel B |
| A+ | Port0[0] | Port0[8] | Port1[0] | Port1[8] |
| A- | Port0[1] | Port0[9] | Port1[1] | Port1[9] |
| B+ | Port0[2] | Port0[10] | Port1[2] | Port1[10] |
| B- | Port0[3] | Port0[11] | Port1[3] | Port1[11] |
| Index + | Port0[4] | Port0[12] | Port1[4] | Port1[12] |
| Index - | Port0[5] | Port0[13] | Port1[5] | Port1[13] |

## INCENCn_ID

ID register to identify this block.

| 15 | 0 |
|---|---|
| ID_Register | |
| R | |

| Field | Description |
|---|---|
| ID_Register15:0] | Value of 0x0002 indicates Dual Incremental Encoder |

## INCENCn_INT

This register provides the status and enables for the encoder interrupts.

| 15 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|
| Reserved | STAT_B_NEG | STAT_B_POS | STAT_A_NEG | STAT_A_POS |
| R,+0 | RC,+0 | RC,+0 | RC,+0 | RC,+0 |

| 7 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|
| Reserved | ENA_B_NEG | ENA_B_POS | ENA_A_NEG | ENA_A_POS |
| R,+0 | RW,+0 | RW,+0 | RW,+0 | RW,+0 |

| Field | Description |
|---|---|
| STAT_B_NEG | Indicates channel B has transitioned from 0x0000 to 0xFFFF. (Negative rollover)<br>'0' = Interrupt has not occurred<br>'1' = Interrupt has occurred.<br>Write '1' to clear. |
| STAT_B_POS | Indicates channel B has transitioned from 0xFFFF to 0x0000. (Positive rollover)<br>'0' = Interrupt has not occurred<br>'1' = Interrupt has occurred.<br>Write '1' to clear. |
| STAT_A_NEG | Indicates channel A has transitioned from 0x0000 to 0xFFFF. (Negative rollover)<br>'0' = Interrupt has not occurred<br>'1' = Interrupt has occurred.<br>Write '1' to clear. |
| STAT_A_POS | Indicates channel A has transitioned from 0xFFFF to 0x0000. (Positive rollover)<br>'0' = Interrupt has not occurred<br>'1' = Interrupt has occurred.<br>Write '1' to clear. |
| ENA_B_NEG | Enables interrupt when channel B transitions from 0x0000 to 0xFFFF. (Negative rollover)<br>'0' = Interrupt is disabled<br>'1' = Interrupt is enabled. |
| ENA_B_POS | Enables interrupt when channel B transitions from 0xFFFF to 0x0000. (Positive rollover)<br>'0' = Interrupt is disabled<br>'1' = Interrupt is enabled. |
| ENA_A_NEG | Enables interrupt when channel A transitions from 0x0000 to 0xFFFF. (Negative rollover)<br>'0' = Interrupt is disabled<br>'1' = Interrupt is enabled. |
| ENA_A_POS | Enables interrupt when channel A transitions from 0xFFFF to 0x0000. (Positive rollover)<br>'0' = Interrupt is disabled<br>'1' = Interrupt is enabled. |

## INCENCn_CLK

This register selects the clock source for sampling the encoder inputs.

| 15 … 4 | 3 … 0 |
|---|---|
| Reserved | CLOCK_SEL[3:0] |
| R,+0 | RW,+0 |

| Field | Description |
|---|---|
| CLOCK_SEL[3:0] | Selects the master clock. Value definitions are:<br>15 Inverted Strobe2<br>14 Inverted Strobe1<br>13 Strobe2<br>12 Strobe1<br>11 Prog. Clock 3<br>10 Prog. Clock 2<br>9 Prog. Clock 1<br>8 Prog. Clock 0<br>7 82C54 TC B2<br>6 82C54 TC B1<br>5 82C54 TC B0<br>4 82C54 TC A2<br>3 82C54 TC A1<br>2 82C54 TC A0<br>1 Reserved<br>0 25 MHz |

## INCENCn_MODE

This register selects the mode of operation for the Incremental Encoder.

| 15 … 8 |
|---|
| PHASE_FLT[7:0] |
| RW,+0 |

| 7 … 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|
| Reserved | DIFF | FILTER | JOIN | IDX_EN | HOLD | ENA |
| R,+0 | RW,+0 | RW,+0 | RW,+0 | RW,+0 | RW,+0 | RW,+0 |

| Field | Description |
|---|---|

<table>
<tr><th>Field</th><th>Description</th></tr>
<tr><td>PHASE_FLT[7:0]</td><td>Phase Filter: Selects if a particular state transition will cause the encoder counter to change. For each bit:<br>’0’ = Transition will change counter<br>’1’ = Transition will not change counter.<br>The bit assignments for the transitions are:
<table>
<tr><th>Bit</th><th>Previous State [B:A]</th><th>Current State [B:A]</th><th>Direction</th></tr>
<tr><td>7</td><td>00</td><td>10</td><td>Down</td></tr>
<tr><td>6</td><td>10</td><td>11</td><td>Down</td></tr>
<tr><td>5</td><td>11</td><td>01</td><td>Down</td></tr>
<tr><td>4</td><td>01</td><td>00</td><td>Down</td></tr>
<tr><td>3</td><td>10</td><td>00</td><td>Up</td></tr>
<tr><td>2</td><td>11</td><td>10</td><td>Up</td></tr>
<tr><td>1</td><td>01</td><td>11</td><td>Up</td></tr>
<tr><td>0</td><td>00</td><td>01</td><td>Up</td></tr>
</table></td></tr>
<tr><td>DIFF</td><td>Selects single-ended or differential mode<br>’0’ = Single Ended. Only “+” inputs are used<br>’1’ = Pseudo-Differential</td></tr>
<tr><td>FILTER</td><td>Enable the input filter<br>’0’ = Filter is disabled<br>’1’ = Filter is enabled</td></tr>
<tr><td>JOIN</td><td>Used to join the two channels into a single 32 bit counter. When the channels are joined, only the Channel A inputs are used.<br>’0’ = Channels are independent<br>’1’ = Channels are joined.</td></tr>
<tr><td>IDX_EN</td><td>Index Enable: When enabled, a high input on the Index input clears the counter<br>’0’ = Index Inputs Disabled<br>’1’ = Index Input Enabled</td></tr>
<tr><td>HOLD</td><td>Register Hold: When enabled, the encoder continues counting in the background, but the VALUE registers remain constant.<br>’0’ = VALUE registers are not held<br>’1’ = VALUE registers are held</td></tr>
<tr><td>ENA</td><td>Enable for this incremental encoder<br>’0’ = Encoder is disabled<br>’1’ = Encoder is enabled</td></tr>
</table>

## INCENCn_VALUEy

Returns the current value of this incremental encoder channel. When INCENCx_MODE[JOIN] = 1, INCENCx_VALUEB contains the most significant word, and INCENCx_VALUEA contains the least significant word. A 16 bit read should be used to read this register when not joined (INCENCx_MODE[JOIN] = 0), and a 32 bit read should be used when joined (INCENCx_MODE[JOIN] = 1). Otherwise, the value can change between read operations. Another option is to set INCENCx_MODE[HOLD] = 1, read the contents of the register, and then set INCENCx_MODE[HOLD] = 0.

This register can only be written to when INCENCx_MODE[ENA] = 0. This allows the counter to be pre-loaded with a known position value.

| 15 | 0 |
|---|---|
| VALUE[15:0] | |
| R(W),+0 | |

| Field | Description |
|---|---|
| VALUE[15:0] | The current value of this incremental encoder channel. |

## Quad Pulse Width Modulator n

The Pulse Width Modulator block provides four PWM outputs. Each output consists of a non-inverted and inverted signal. These signals are available on select pins as peripheral outputs. The period and width of the output is set with 16 bit resolution.

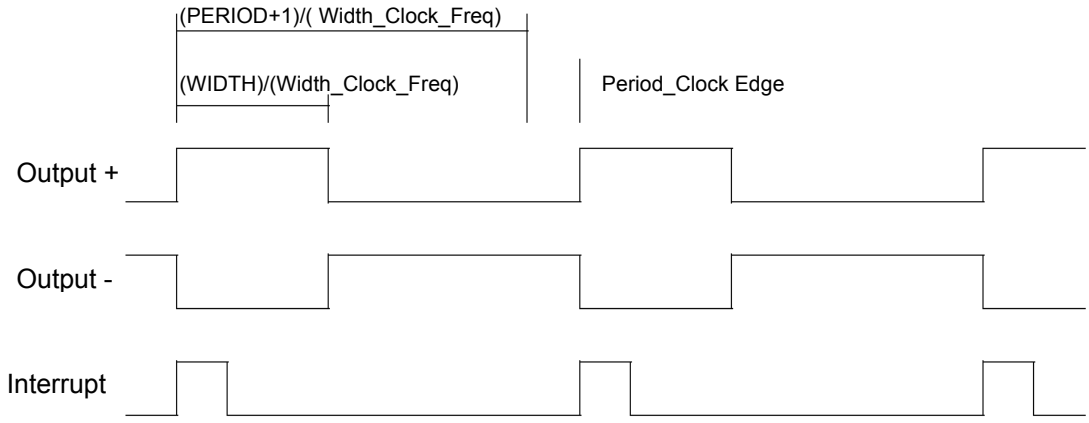



**Figure 6: PWM Output**

All of the PWM outputs have the same period. The pulse width of each of the four outputs is individually adjustable.

The PWM can use separate clocks for width and period. The width clock is used to decrement the counter. When the counter reaches zero, it will wait for the next period clock to re-load the counter with the period value.

In a typical PWM implementation, the same clock is used for width and period. By using separate clocks, a high resolution can be achieved with low duty cycle outputs. For example, if a 1 MHz clock is used for the period clock and the PERIOD register is set to its maximum value, and a 10 MHz clock is used for the width clock, the duty cycle range is 0% to 10%, with a full 16 bit resolution across that range.

Note that if the PERIOD register is set to its maximum value, a duty cycle of 100% cannot be achieved.

An interrupt is generated at the beginning of every period.

The width register is checked at the beginning of every period. If the width register is modified in the middle of a period, the output will not be affected until the next period.

## PWMn_ID

ID register to identify this block.

| 15 | 0 |
|---|---|
| ID_Register | |
| R | |

| Field | Description |
|---|---|
| ID_Register15:0] | Value of 0x0003 indicates Pulse Width Modulator |

## PWMn_MODE

This register is used to enable and disable the Pulse Width Modulator. When disabled, all non-inverted outputs are low, and all inverted outputs are high, and interrupts are not generated.

| 15 | 1 | 0 |
|---|---|---|
| Reserved | | ENA |
| R,+0 | | RW,+0 |

| Field | Description |
|---|---|
| ENA | Enables or disabled the PWM.<br>'0' = Disabled<br>'1' = Enabled |

## PWMn_CLK

This register selects the clock sources for the period and width of the PWM output.

| 15 | 8 | 7 | 4 | 3 | 0 |
|---|---|---|---|---|---|
| Reserved | | PER_CLK[3:0] | | WIDTH_CLK[3:0] | |
| R,+0 | | RW,+0 | | RW,+0 | |

| Field | Description |
|---|---|
| PER_CLK [3:0] | Selects the master clock for the period counter. Value definitions are:<br>15 Inverted Strobe2<br>14 Inverted Strobe1<br>13 Strobe2<br>12 Strobe1<br>11 Prog. Clock 3<br>10 Prog. Clock 2<br>9 Prog. Clock 1<br>8 Prog. Clock 0<br>7 82C54 TC B2<br>6 82C54 TC B1<br>5 82C54 TC B0<br>4 82C54 TC A2<br>3 82C54 TC A1 |

| Field | Description |
|---|---|
| | 2 82C54 TC A0<br>1 Reserved<br>0 25 MHz |
| WIDTH_CLK[3:0] | Selects the master clock for the width counter. See above for value definitions. |

### PWMn_PERIOD

Sets the maximum width of the PWM outputs. If the period clock and width clock are the same (PWMn_CLK[PER_CLK] = PWMn_CLK[WIDTH_CLK]), this will also set the PWM period. See Figure 6 on page 53 for more details.

| 15 | 0 |
|---|---|
| PERIOD[15:0] | |
| RW,+0 | |

| Field | Description |
|---|---|
| PERIOD[15:0] | The period of the output is the next period clock after:<br>$\frac{(PERIOD+1)}{Width\_Clock\_Frequency}$ |

### PWMn_WIDTHx

Sets the width of output x of the pulse width modulator. The width is based on the clock selected in PWMn_CLK[WIDTH_CLK]. The width is defined as the time that the non-inverted output is high, and the inverted output is low.

The width register is checked at the beginning of every period. If the width register is modified in the middle of a period, the output will not be affected until the next period.

Note that with PWMn_PERIOD set to the maximum value, and the period clock and width clock set to the same source, a 100% duty cycle is not possible.

| 15 | 0 |
|---|---|
| WIDTH[15:0] | |
| RW,+0 | |

| Field | Description |
|---|---|
| WIDTH[15:0] | The width of the output:<br>$\frac{WIDTH}{Width\_Clock\_Frequency}$ |

## 82C54 Timer Counter n

The following section is taken from the MSM82C54 Datasheet from Oki Semiconductors. For information on programming the 82C54 timer counters, please consult the datasheet.

## DESCRIPTION OF OPERATION

MSM82C54-2 functions are selected by control words from the CPU. In the required program sequence, the control word setting is followed by the count value setting and execution of the desired timer operation.

### Control Word and Count Value Program

Each counter operating mode is set by control word programming. The control word format is outlined below.

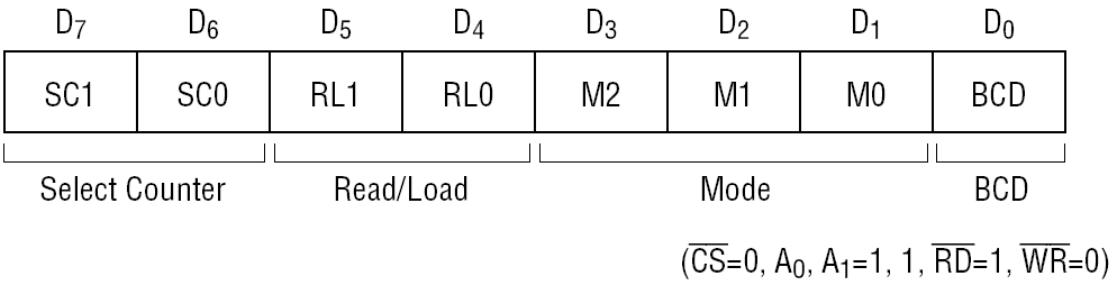



**Table 7: Select Counter SC[1:0]: Selection of set counter**

| SC[1:0] | Set Contents |
|---|---|
| 00 | Counter #0 Selection |
| 01 | Counter #1 Selection |
| 10 | Counter #2 Selection |
| 11 | Read Back Command |

**Table 8: Read/Load RL[1:0]: Count Value Reading/Loading format setting**

| RL[1:0] | Set Contents |
|---|---|
| 00 | Counter Latch Operation |
| 01 | Reading/Loading of Least Significant Byte (LSB) |
| 10 | Reading/Loading of Most Significant Byte (MSB) |
| 11 | Reading/Loading of LSB followed by MSB |

**Table 9: Mode M[2:0]: Operation waveform mode setting**

| M[1:0] | Set Contents | Min Count Value | Max Count Value |
|---|---|---|---|
| 000 | Mode 0 (Interrupt on Terminal Count) | 1 | 0 |
| 001 | Mode 1 (Programmable One-Shot) | 1 | 0 |
| x10 | Mode 2 (Rate Generator) | 2 | 0 |
| x11 | Mode 3 (Square Wave Generator) | 2 | 0 |
| 100 | Mode 4 (Software Triggered Strobe) | 1 | 0 |
| 101 | Mode 5 (Hardware Triggered Strobe) | 1 | 0 |

x denotes “not specified”
Count value of 0 executed 0x10000 count

**Table 10: BCD: Operation count mode setting**

| BCD | Set Contents |
|---|---|
| 0 | Binary Count (16-bit Binary) |
| 1 | BCD Count (4-decade Binary Coded Decimal) |

After setting Read/Load, Mode and BCD in each counter as outlined above, next set the desired count value. (In some Modes, the count value is set first. In next clock, loading is performed, and then counting starts.) This count value setting must conform to the Read/Load format set in advance. Note that the internal counters are reset to 0000H during control word setting. The counter value (0000H) can't be read.

The program sequence of the MSM82C54-2 is flexible. Free sequence programming is possible as long as the two following rules are observed:

(i) Write the control word before writing the initial count value in each counter.

(ii) Write the initial count value according to the count value read/write format specified by the control word.

**Note:** Unlike the MSM82C53-2, the MSM82C54-2 allows count value setting for another counter between LSB and MSB settings.

## Mode definition

### *Mode 0*

- Application: Event counter
- Output operation: The output is set to "L" level by the control word setting, and kept at "L" level until the counter value becomes 0.
- Gate function: "H" level validates the count operation, and "L" level invalidates it. The gate does not affect the output.
- Count value load timing: after the control word and initial count value are written, the count value is loaded to the CE at the falling edge of the next clock pulse. The first clock pulse does not cause the count value to be decremented. In other words, if the initial count value is N, the output is not set to "H" level until the input of (N+1) the clock pulse after the initial count value writing.
- Count value writing during counting: The count value is loaded in the CE at the falling edge of the next clock, and counting with the new count value continues. The operation for 2-byte count is as follows:
    - The counting operation is suspended when the first byte is written. The output is immediately set to "L" level. (No clock pulse is required.)
    - After the second byte is written, the new count value is loaded to the CE at the falling edge of the next clock.
    - For the output to go to "H" level again, N+1 clock pulse are necessary after new count value N is written.
- Count value writing when the gate signal is "L" level: The count value is also loaded to the CE at the falling edge of the next clock pulse in this case. When the gate signal is set

to “H” level, the output is set to “H” level after the lapse of N clock pulses. Since the count value is already loaded in the CE, no clock pulse for loading in the CE is necessary.

### *Mode 1*

- Application: Digital one-shot
- Output operation: The output is set to “H” level by the control word setting. It is set to “L” level at the falling edge of the clock succeeding the gate trigger, and kept at “L” level until the counter value becomes 0. Once the output is set to “H” level, it is kept at “H” level until the clock pulse succeeding the next trigger pulse.
- Count value load timing: After the control word and initial count value are written, the count value is loaded to the CE at the falling edge of the clock pulse succeeding the gate trigger and set the output to “L” level. The one-shot pulse starts in this way. If the initial count value is N, the one-shot pulse interval equals N clock pulses. The one-shot pulse is not repetitive.
- Gate function: The gate signal setting to “L” level after the gate trigger does not affect the output. When it is set to “H” level again from “L” level, gate retriggering occurs, the CR count value is loaded again, and counting continues.
- Count value writing during counting: It does not affect the one-shot pulse being counted until retriggering occurs.

### *Mode 2*

- Application: Rate generator, real-time interrupt clock.
- Output operation: The output is set to “H” level by control word setting. When the initial count value is decremented to 1, the output is set to “L” level during one clock pulse, and is then set to “H” level again. The initial count value is reloaded, and the above sequence repeats. In mode 2, the same sequence is repeated at intervals of N clock pulses if the initial count value is N for example.
- Gate function: “H” level validates counting and “L” level invalidates it. If the gate signal is set to “L” level when the output pulse is “L” level, the output is immediately set to “H” level. At the falling edge of the clock pulse succeeding the trigger, the count value is reloaded and counting starts. The gate input can be used for counter synchronization in this way.
- Count value load timing: After the control word and initial count value is written, the count value is loaded to the CE at the falling edge of the next clock pulse. The output is set to “L” level upon lapse of N clock pulses after writing the initial count value N. Counter synchronization by software is possible in this way.
- Count value writing during counting: Count value writing does not affect the current counting operation sequence. If new count value writing completes and the gate trigger arrives before the end of current counting operation, the count value is loaded to the CE at the falling edge of next clock pulse and counting continues from the new count value. If no gate trigger arrives, the new count value is loaded to the CE at the end of the current counting operation cycle. In mode 2, count value of 1 is prohibited.

### *Mode 3*

- Application: Baud rate generator, square wave generator

- Output operation: Same as mode 2 except that the output duty is different. The output is set to “H” level by control word setting. When the count becomes half the initial count value, the output is set to “L” level and kept at “L” level during the remainder of the count. Mode 3 repeats the above sequence periodically. If the initial count value is N, the output becomes a square wave with a period of N.

- Gate operation: “H” level validates counting and “L” level invalidates it. If the gate signal is set to “L” level when the output is “L” level, the output is immediately set to “H” level. The initial count value is reloaded at the falling edge of the clock pulse succeeding the next gate trigger. The gate can be used for counter synchronization in this way.

- Count value load timing: After the control word and initial count value are written, the count value is loaded to the CE at the falling edge of the next clock pulse; Counter synchronization by software is possible in this way.

- Count value writing during counting: The count value writing does not affect the current counting operation. When the gate trigger input arrives before the end of a half cycle of the square wave after writing the new count value, the new count value is loaded in the CE at the falling edge of the next clock pulse, and counting continues using the new count value. If there is no gate trigger, the new count value is loaded at the end of the half cycle and counting continues.

- Even number counting operation: The output is initially set to “H” level. The initial count value is loaded to the CE at the falling edge of the next clock pulse, and is decremented by 2 by consecutive clock pulses. When the counter value becomes 2, the output is set to “L” level, the initial value is reloaded and then the above operation is repeated.

- Odd number counting operation: The output is initially set to “H” level. At the falling edge of the next clock pulse, the initial count value minus one is loaded in the CE, and then the value is decremented by 2 by consecutive clock pulses. When the counter value becomes 0, the output is set to “L” level, and then the initial count value minus 1 is reloaded to the CE. The value is then decremented by 2 by consecutive clock pulses. When the counter value becomes 2, the output is again set to “H” level and the initial count value minus 1 is again reloaded. The above operations are repeated. In other words, the output is set to “H” level during (N+1)/2 counting and to “L” level during (N-1)/2 counting in the case of odd number counting.

### *Mode 4*

- Application: Software trigger strobe

- Output operation: The output is initially set to “H” level. When the counter value becomes 0, the output goes to “L” level during one clock pulse, and then restores “H” level again. The count sequence starts when the initial count value is written.

- Gate function: “H” level validates counting and “L” level invalidates counting. The gate signal does not affect the output.

- Count value load timing: After the control word and initial count value are written, the count value is loaded to the CE at the falling edge of the next clock pulse. The clock pulse does not decrement the initial count value. If the initial count value is N, the strobe is not output unless N+1 clock pulses are input after the initial count value is written,

- Count value writing during counting: The new count value is written to the CE at the falling edge of the next clock pulse, and counting continues using the new count value. The operation for 2-byte count is as follows:

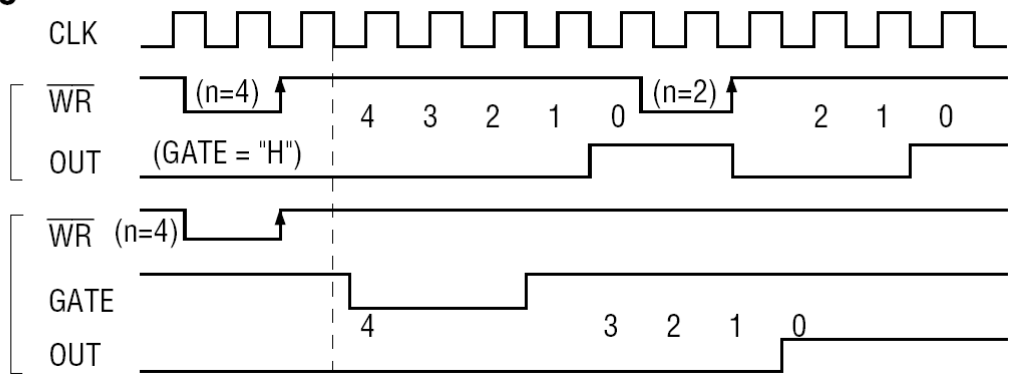
- 1) First byte writing does not affect the counting operation.
- 2) After the second byte is written, the new count value is loaded to the CE at the falling edge of the next clock pulse.
- This means that the counting operation is retriggered by software. The output strobe is set to “L” level upon input of N+1 clock pulses after the new count value N is written.

## *Mode 5*

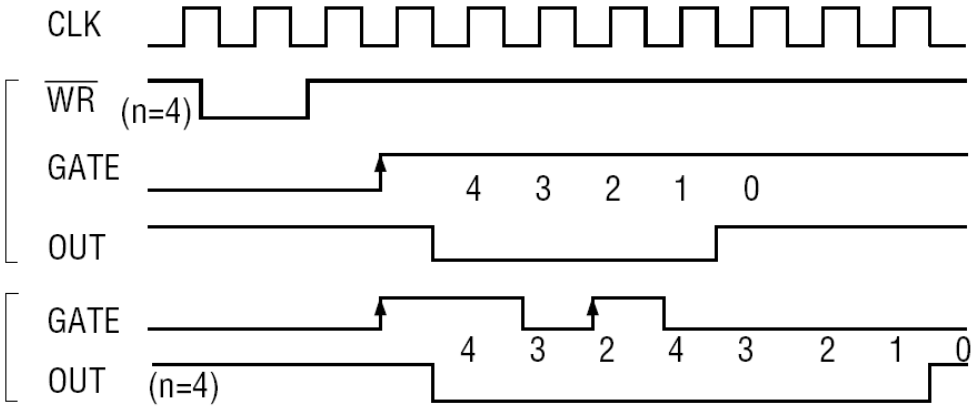
- Application: Hardware trigger strobe
- Output operation: The output is initially set to “H” level. When the counter value becomes 0 after triggering by the rising edge of the gate pulse, the output goes to “L” level during one clock pulse, and then restores “H” level.
- Count value load timing: Even after the control word and initial count value are written, loading to the CE does not occur until the input of the clock pulse succeeding the trigger. For the clock pulse for CE loading, the count value is not decremented. If the initial count value is N, therefore, the output is not set to “L” level until N+1 clock pulses are input after triggering.
- Gate function: The initial count value is loaded to the CE at the falling edge of the clock pulse succeeding gate triggering. The count sequence can be retriggered. The gate pulse does not affect the output.
- Count value writing during counting: The count value writing does not affect the current counting sequence. If the gate trigger is generated after the new count value is written and before the current counting ends, the new count value is loaded to the CE at the falling edge of the next clock pulse, and counting continues using the new count value. The various roles of the gate input signals in the above modes are summarized in the following table.

| Mode \ Gate | "L" Level Falling Edge | Rising Edge | "H" Level |
|---|---|---|---|
| 0 | Counting not possible | | Counting possible |
| 1 | | (1) Start of counting<br>(2) Retriggering | |
| 2 | (1) Counting not possible<br>(2) Counter output forced to "H" level | Start of counting | Counting possible |
| 3 | (1) Counting not possible<br>(2) Counter output forced to "H" level | Start of counting | Counting possible |
| 4 | Counting not possible | | Counting possible |
| 5 | | (1) Start of counting<br>(2) Retriggering | |

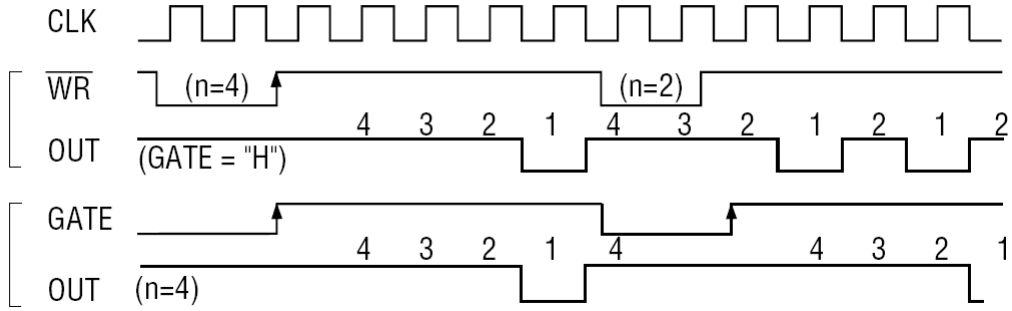
## Mode 0



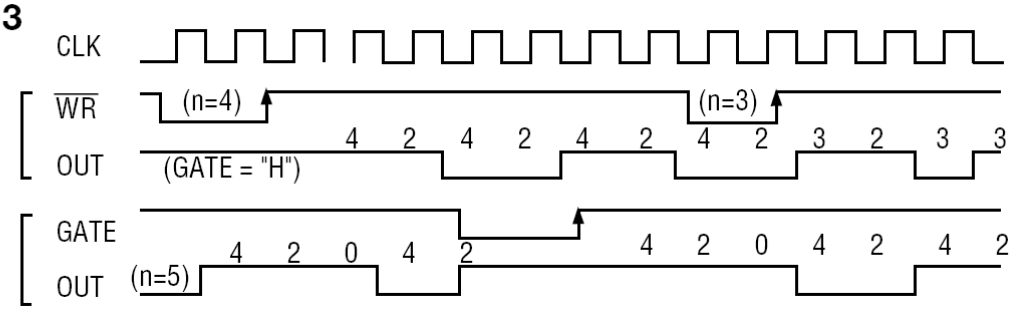

## Mode 1



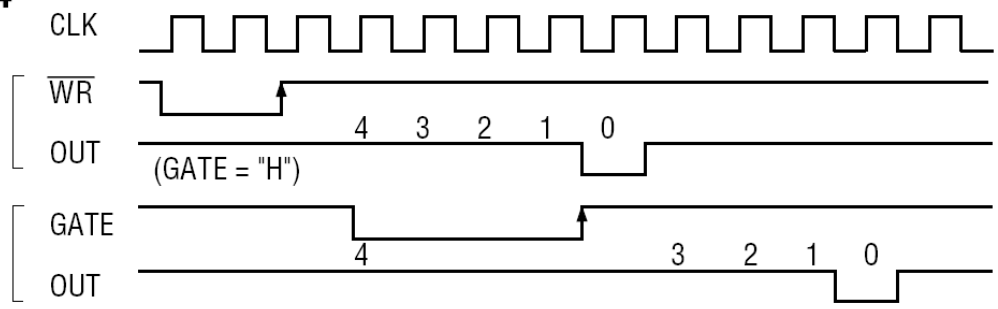

## Mode 2



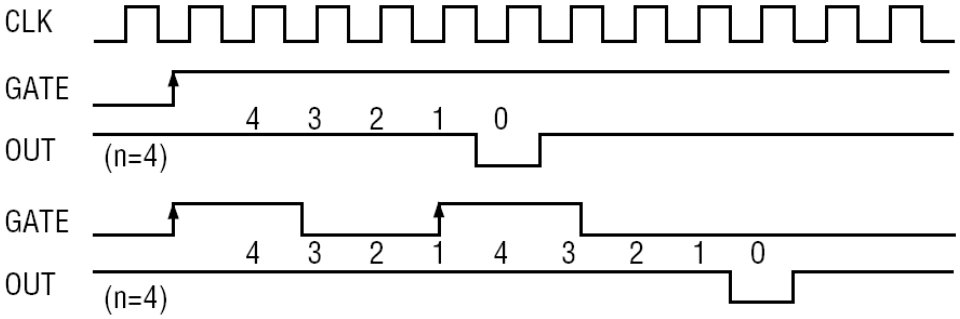

## Mode 3



**Mode 4**




**Mode 5**




**Note :** "n" is the value set in the counter.
Figures in these diagrams refer to counter values.

### Reading Counter Values

All MSM82C54-2 counting is down-counting, the counting being in steps of 2 in mode 3. Counter values can be read during counting by: (1) direct reading, (2) counter latching (“read on the fly”), and (3) read back command.

#### *Direct reading*

Counter values can be read by direct reading operations. Since the counter value read according to the timing of the RD and CLK signals is not guaranteed, it is necessary to stop the counting by a gate input signal, or to interrupt the clock input temporarily by an external circuit to ensure that the counter value is correctly read.

#### *Counter latching*

In this method, the counter value is latched by writing counter latch command, thereby enabling a stable value to be read without effecting the counting in any way at all. The output latch (OL) of the selected counter latches the count value when a counter latch command is written. The count value is held until it is read by the CPU or the control word is set again.

If a counter latch command is written again before reading while a certain counter is latched, the second counter latch command is ignored and the value latched by the first counter latch command is maintained.

The MSM82C54-2 features independent reading and writing from and to the same counter. When a counter is programmed for the 2-byte counter value, the following sequence is possible:

1. Count value (LSB) reading
2. New count value (LSB) writing
3. Count value (MSB) reading
4. New count value (MSB) writing

An example of a counter latching program is given below.

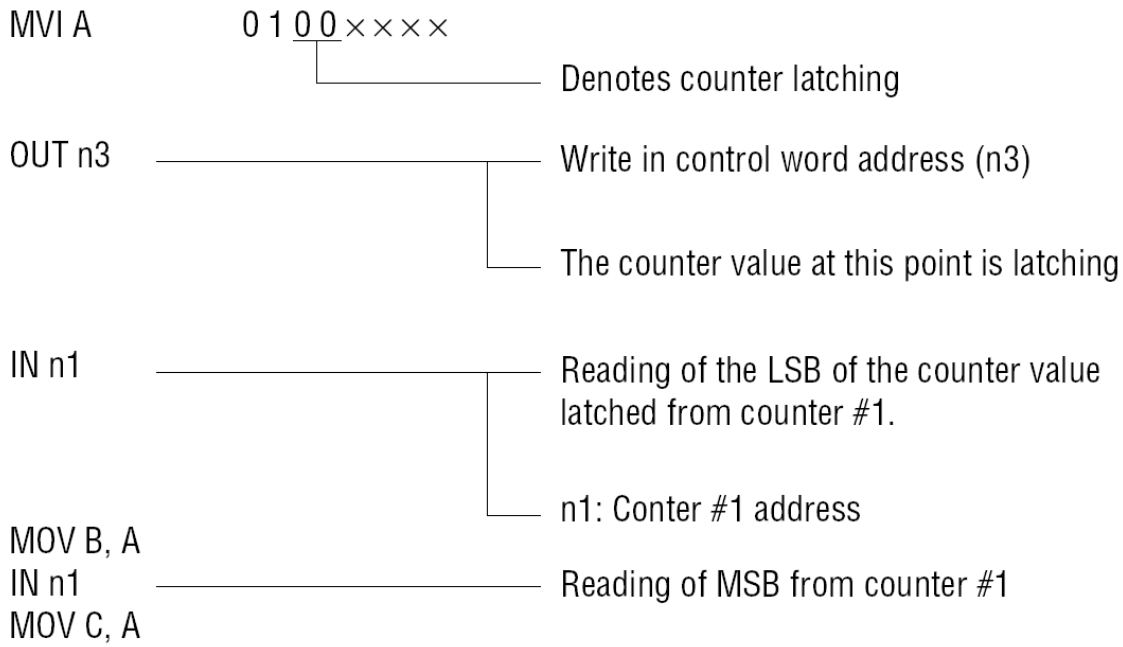

MVI A
0 1 0 0 × × × ×
Denotes counter latching
OUT n3
Write in control word address (n3)
The counter value at this point is latching
IN n1
Reading of the LSB of the counter value latched from counter #1.
n1: Conter #1 address
MOV B, A
IN n1
Reading of MSB from counter #1
MOV C, A

**Figure 7: Counter latching executed for counter #1 (Read/Load 2-byte setting)**

## *Read Back Command Operation*

Use of the read back command enables the user to check the count value, program mode, output pin state and null count flag of the selected counter. The command is written in the control word register, and the format is as shown below. For this command, the counter selection occurs according to bits D3, D2 and D1.

| $D_7$ | $D_6$ | $D_5$ | $D_4$ | $D_3$ | $D_2$ | $D_1$ | $D_0$ |
|---|---|---|---|---|---|---|---|
| 1 | 1 | $\overline{\text{COUNT}}$ | $\overline{\text{STATUS}}$ | CNT2 | CNT1 | CNT0 | 0 |

($CS=0$, $A_0$, $A_1=1$, $RD=1$, $WR=0$)

$D_5$: 0 = Selected counter latch operation

$D_4$: 0 = Selected counter status latch operation

$D_3$: 1 = Counter #2 selection

$D_2$: 1 = Counter #1 selection

$D_1$: 1 = Counter #0 selection

$D_0$: 0 Fixed

It is possible to latch multiple counters by using the read back command. Latching of a read counter is automatically canceled but other counters are kept latched. If multiple read back commands are written for the same counter, commands other than the first one are ignored. It is also possible to latch the status information of each counter by using the read back command. The status of a certain counter is read when the counter is read. The counter status format is as follows:

Bits D5 to D0 indicate the mode programmed by the most recently written control word.

Bit D7 indicates the status of the output pin. Use of this bit makes it possible to monitor the counter output, so the corresponding hardware may be omitted.

| $D_7$ | $D_6$ | $D_5$ | $D_4$ | $D_3$ | $D_2$ | $D_1$ | $D_0$ |
|---|---|---|---|---|---|---|---|
| OUTPUT | NULL COUNT | RL1 | RL0 | M2 | M1 | M0 | BCD |

$D_7$: 1 = Output pin status is 1.
0 = Output pin status is 0.

$D_6$: 1 = Null count
0 = Count value reading is effective

$D_5$ - $D_0$: Programmed mode of counter
(See the control word format.)

Null count indicates the count value finally written in the counter register (CR) has been loaded in the counter element (CE). The time when the count value was loaded in the CE depends on the mode of each counter, and it cannot be known by reading the counter value because the count

value does not tell the new count value if the counter is latched. The null count operation is shown below.

| Operation | Result |
|---|---|
| A. Control word register writing | Null count = 1 |
| B. Count register (CR) writing | Null count = 1 |
| C. New count loading to CE (CR->CE) | Null count = 0 |

Note: The null count operation for each counter is independent. When the 2-byte count is programmed, the null count is set to 1 when the count value of the second byte is written.

If status latching is carried out multiple times before status reading, other than the first status latch is ignored.

Simultaneous latching of the count and status of the selected counter is also possible. For this purpose, set bits D4 and D3, COUNT and STATUS bits, to 00. This is functionally the same as writing two separate read back commands at the same time. If counter/status latching is carried out multiple times before each reading, other than the first one is ignored here again. The example is shown below.

| Command | | | | | | | | Contents | Counter 0 | | Counter 1 | | Counter 2 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $D_7$ | $D_6$ | $D_5$ | $D_4$ | $D_3$ | $D_2$ | $D_1$ | $D_0$ | | Count | Status | Count | Status | Count | Status |
| 1 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | Read back status and count (counter 0) | L | L | — | — | — | — |
| 1 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | Read back status (counter 1) | L | L | — | L | — | — |
| 1 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | Read back status (counter 1 and 2) | L | L | — | L (NOTE) | — | L |
| 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | Read back status (counter 2) | L | L | — | L | L | L |
| 1 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | Read back status and count (counter 1) | L | L | L | L (NOTE) | L | L |
| 1 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | Read back status (counter 0) | L | L (NOTE) | L | L | L | L |

Note: The latch command at this time point is ignored, and the first latch command is valid.

If both the count and status are latched, the status latched in the first counter read operation is read. The order of count latching and status latching is irrelevant. The count(s) of the next one or two reading operations is or are read.

# *PLX Registers*

The PLX9056 PCI Accelerator on the DM7820/DM9820 contains several registers to control interrupts and the two DMA engines. These engines allow data to be transferred on-demand with no load on the processor. The following sections describe the registers used for programming the DMA engines. This information is taken from PLX PCI9065BA Datasheet. For more information, please consult the datasheet.

## Memory Map Overview

Table 11 shows the memory map of the DM7820/DM9820 DMA registers. These are found at the memory offset from BAR0, or the I/O offset from BAR1.

**Table 11: PLX DM7820/DM9820 Memory Map**

| Offset (Hex) | | Register Name | Register Description |
|---|---|---|---|
| **DMA Channel 0** | | | |
| **0x80** | | DMAMODE0 | DMA Channel 0 Mode |
| **0x84** | **0x88** | DMAPADR0 | DMA Channel 0 PCI Address |
| **0x88** | **0x8C** | DMALADR0 | DMA Channel 0 Local Address |
| **0x8C** | **0x84** | DMASIZ0 | DMA Channel 0 Transfer Size (Bytes) |
| **0x90** | | DMADPR0 | DMA Channel 0 Descriptor Pointer |
| **DMA Channel 1** | | | |
| **0x94** | | DMAMODE1 | DMA Channel 1 Mode |
| **0x98** | **0x9C** | DMAPADR1 | DMA Channel 1 PCI Address |
| **0x9C** | **0xA0** | DMALADR1 | DMA Channel 1 Local Address |
| **0xA0** | **0x98** | DMASIZ1 | DMA Channel 1 Transfer Size (Bytes) |
| **0xA4** | | DMADPR1 | DMA Channel 1 Descriptor Pointer |
| **Command and Status** | | | |
| **0xA8** | | DMACSR0 | DMA Channel 0 Command/Status |
| **0xA9** | | DMACSR1 | DMA Channel 1 Command/Status |
| **0xAC** | | DMAARB | DMA Arbitration |
| **0xB0** | | DMATHR | DMA Threshold |
| **0xB4** | | DMADA0 | DMA Channel 0 PCI Dual Address Cycle Upper Address |
| **0xB8** | | DMADA1 | DMA Channel 1 PCI Dual Address Cycle Upper Address |
| **Interrupt** | | | |
| **0x68** | | INTCSR | Interrupt Control/Status |

Where two addresses are given, the left column is the address when DMAMODEn[20] =0, and the right column is the address when DMAMODEn[20] =1.

# DMA Register Description

## DMAMODEn

DMA Mode

| Bit | Description | Read | Write | Value after Reset | Value to Use |
|---|---|---|---|---|---|
| 1:0 | **Local Bus Data Width.** Writing of the following values indicates the associated bus data width:<br>00b = 8 bit<br>01b = 16 bit<br>10b or 11b = 32 bit | Yes | Yes | 11b | 11b |
| 5:2 | **Internal Wait State Counter** (Address-to-Data; Data-to-Data; 0 to 15 Wait States). | Yes | Yes | 0h | 0h |
| 6 | **TA#/READY# Input Enable.** Writing 1 enables READY# input. Writing 0 disables READY# input. | Yes | Yes | 1 | 1 |
| 7 | **Continuous Burst Enable.** When bursting is enabled (DMAMODE0[8]=1), writing 1 enables Continuous Burst mode and writing 0 enables Burst-4 mode. Writing 1 additionally enables BTERM# input, which when asserted overrides the READY# input state (if READY# is enabled, DMAMODE0[6]=1).<br>**Notes:** This bit is referred to as the "BTERM# Input Enable" bit. Refer to Section 4.2.5 of the PCI9056 datasheet for further details. | Yes | Yes | 0 | 1 |
| 8 | **Local Burst Enable.** Writing 1 enables Local bursting. Writing 0 disables Local bursting. | Yes | Yes | 0 | 1 |
| 9 | **Scatter/Gather Mode.** Writing 1 indicates DMA Scatter/Gather mode is enabled. For Scatter/Gather mode, the DMA source and destination addresses and byte count are loaded from memory in PCI or Local Address spaces. Writing 0 indicates DMA Block mode is enabled. | Yes | Yes | 0 | x |
| 10 | **Done Interrupt Enable.** Writing 1 enables an interrupt when done. Writing 0 disables an interrupt when done. If DMA Clear Count mode is enabled (DMAMODE0[16]=1), the interrupt does not occur until the byte count is cleared | Yes | Yes | 0 | x |
| 11 | **Local Addressing Mode.** Writing 1 holds the Local Address Bus constant. Writing 0 indicates the Local Address is incremented. | Yes | Yes | 0 | 1 |

| | | | | | |
|---|---|---|---|---|---|
| 12 | **Demand Mode.** Writing 1 causes the DMA Controller to operate in Demand mode. In Demand mode, the DMA Controller transfers data when its DREQ0# input is asserted. Asserts DACK0# to indicate the current Local Bus transfer is in response to DREQ0# input. The DMA Controller transfers Lwords (32 bits) of data. This may result in multiple transfers for an 8- or 16-bit bus. | Yes | Yes | 0 | 1 |
| 13 | **Memory Write and Invalidate Mode for DMA Transfers.** When set to 1, the PCI 9056 performs Memory Write and Invalidate cycles to the PCI Bus. The PCI 9056 supports Memory Write and Invalidate sizes of 8 or 16 Lwords. The size is specified in the System Cache Line Size bits (PCICLSR[7:0]). If a size other than 8 or 16 is specified, the PCI 9056 performs Write transfers, rather than Memory Write and Invalidate transfers. Transfers must start and end at cache line boundaries. PCICR[4] must be set to 1. | Yes | Yes | 0 | x |
| 14 | **EOT# Enable.** Writing 1 enables the EOT# input pin. Writing 0 disables the EOT# input pin. If DMAMODE0[14] and DMAMODE1[14]=00b, the EOT# pin becomes the DMPAF pin. | Yes | Yes | 0 | 0 |
| 15 | **Fast/Slow Terminate Mode Select.** Writing 0 sets the PCI 9056 into Slow Terminate mode. As a result, BLAST# is asserted on the last Data transfer to terminate the DMA transfer. Writing 1 sets the PCI 9056 into Fast Terminate mode, and indicates the PCI 9056 DMA transfer terminates immediately when EOT# (if enabled) is asserted, or during DMA Demand mode when DREQ0# is de-asserted. | Yes | Yes | 0 | 0 |
| 16 | **Clear Count Mode.** Writing 1 clears the byte count in each Scatter/Gather descriptor when the corresponding DMA transfer is complete. | Yes | Yes | 0 | x |
| 17 | **Interrupt Select.** Writing 1 routes the interrupt to the PCI interrupt (INTA#). Writing 0 routes the interrupt to the Local interrupt output (LINTo#). | Yes | Yes | 0 | 1 |
| 18 | **DAC Chain Load.** When set to 1, enables the descriptor to load the PCI Dual Address Cycles value. Otherwise, the descriptor loads the DMADAC0 register contents. | Yes | Yes | 0 | x |
| 19 | **EOT# End Link.** Used only for DMA Scatter/Gather transfers. Value of 1 indicates that when EOT# is asserted, the DMA transfer ends the current Scatter/Gather link and continues with the remaining Scatter/Gather transfers. Value of 0 indicates that when EOT# is asserted, the DMA transfer ends the current Scatter/Gather transfer and does not continue with the remaining Scatter/Gather transfers. | Yes | Yes | 0 | 0 |

| | | | | | |
|---|---|---|---|---|---|
| 20 | **Ring Management Valid Mode Enable.** Value of 0 indicates the Ring Management Valid bit (DMASIZ0[31]) is ignored. Value of 1 indicates the DMA descriptors are processed only when the Ring Management Valid bit is set (DMASIZ0[31]=1). If the Valid bit is set, the transfer count is 0, and the descriptor is not the last descriptor in the chain. The DMA Controller then moves to the next descriptor in the chain.<br>**Note:** Descriptor Memory fields are re-ordered when this bit is set. | Yes | Yes | 0 | x |
| 21 | **Ring Management Valid Stop Control.** Value of 0 indicates the DMA Scatter/Gather controller continuously polls a descriptor with the Valid bit set to 0 (invalid descriptor) if Ring Management Valid Mode is enabled (DMAMODE0[20]=1). Value of 1 indicates the Scatter/Gather controller stops polling when the Ring Management Valid bit with a value of 0 is detected (DMASIZ0[31]=0). In this case, the CPU must restart the DMA Controller by setting the Start bit (DMACSR0[1]=1). A pause clearing the Start bit (DMACSR0[1]=0) sets the DMA Done bit (DMACSR0[4]=1). | Yes | Yes | 0 | x |
| 31:22 | **Reserved** | Yes | No | 0 | 0 |

## DMAPADRn

DMA PCI Address

| Bit | Description | Read | Write | Value after Reset | Value to Use |
|---|---|---|---|---|---|
| 31:0 | **PCI Address.** Indicates from where in PCI Memory space DMA transfers (reads or writes) start. Value is a physical address. | Yes | Yes | 0h | x |

## DMALAPADRn

DMA Local Address

| Bit | Description | Read | Write | Value after Reset | Value to Use |
|---|---|---|---|---|---|
| 31:0 | **DMA Channel Local Address.** Indicates from where in Local Memory space DMA transfers (reads or writes) start. | Yes | Yes | 0h | x |

## DMASIZn

DMA Transfer Size

| Bit | Description | Read | Write | Value after Reset | Value to Use |
|---|---|---|---|---|---|
| 22:0 | **Transfer Size (Bytes).** Indicates the number of bytes to transfer during a DMA operation. | Yes | Yes | 0h | x |
| 30:23 | **Reserved** | Yes | No | 0h | 0h |
| 31 | **Ring Management Valid.** When Ring Management Valid Mode is enabled (DMAMODE0[20]=1), indicates the validity of this DMA descriptor. | Yes | Yes | 0 | x |

## DMAPRn

DMA Channel n Descriptor Pointer

| Bit | Description | Read | Write | Value after Reset | Value to Use |
|---|---|---|---|---|---|
| 0 | **Descriptor Location.** Writing 1 indicates PCI Address space. Writing 0 indicates Local Address space. | Yes | Yes | 0h | 1 |
| 1 | **End of Chain.** Writing 1 indicates end of chain. Writing 0 indicates not end of chain descriptor. (Same as DMA Block mode.) | Yes | Yes | 0h | x |
| 2 | **Interrupt after Terminal Count.** Writing 1 causes an interrupt to be asserted after the terminal count for this descriptor is reached. Writing 0 disables interrupts from being asserted. | Yes | Yes | 0h | x |
| 3 | **Direction of Transfer.** Writing 1 indicates transfers from the Local Bus to the PCI Bus. Writing 0 indicates transfers from the PCI Bus to the Local Bus. | Yes | Yes | 0h | x |
| 31:4 | **Next Descriptor Address.** X0h-aligned (DMADPR0[3:0]=0h). | Yes | Yes | 0h | x |

## DMACSRn

DMA Channel n Command/Status

| Bit | Description | Read | Write | Value after Reset | Value to Use |
|---|---|---|---|---|---|
| 0 | **Enable.** Writing 1 enables the channel to transfer data. Writing 0 disables the channel from starting a DMA transfer, and if in the process of transferring data, suspends the transfer (pause). | Yes | Yes | 0h | 1 |
| 1 | **Start.** Writing 1 causes the channel to start transferring data if the channel is enabled. | Yes | Yes/ Set | 0h | x |

| | | | | | |
|---|---|---|---|---|---|
| 2 | **Abort.** Writing 1 causes the channel to abort the current transfer. The DMA Channel 0 Enable bit must be cleared (DMACSR0[0]=0). Sets the DMA Channel 0 Done bit (DMACSR0[4]=1) when the abort is complete. | Yes | Yes/ Set | 0h | x |
| 3 | **Clear Interrupt.** Writing 1 clears DMA Channel 0 interrupts. | Yes | Yes/ Clr | 0h | x |
| 4 | **Done.** Reading 1 indicates the transfer is complete. The transfer may be complete either because the DMA transfer finished successfully, or that the DMA transfer was aborted when software set the Abort bit (DMACSR0[2]=1). Reading 0 indicates the Channel transfer is not complete. | Yes | No | 0h | x |
| 7:5 | **Reserved.** | Yes | No | 000b | 000b |

## DMAARB

DMA Arbitration

| Bit | Description | Read | Write | Value after Reset | Value to Use |
|---|---|---|---|---|---|
| 18:0 | **Reserved.** | Yes | (Do not Modify) | 0 | 0 |
| 20:19 | **DMA Channel Priority.** Writing 00b indicates a rotational priority scheme. Writing 01b indicates Channel 0 has priority. Writing 10b indicates Channel 1 has priority. Value of 11b is ***reserved***. | Yes | Yes | 00b | 00b |
| 31:21 | **Reserved.** | Yes | (Do not Modify) | 0000 0000 001b | 0000 0011 001b |

## DMATHR

DMA Threshold

| Bit | Description | Read | Write | Value after Reset | Value to Use |
|---|---|---|---|---|---|
| 3:0 | **DMA Channel 0 PCI-to-Local Almost Full (C0PLAF).** Number of full (Lword x 2) entries (plus 1, times 2) in the FIFO before requesting the Local Bus for writes. Nybble values 0h through Eh may be used. (Refer to Table 12.)<br>(15 - C0PLAF) > C0LPAE. | Yes | Yes | 0h | x |

| | | | | | |
|---|---|---|---|---|---|
| 7:4 | **DMA Channel 0 Local-to-PCI Almost Empty (C0LPAE).** Number of empty (Lword x 2) entries (plus 1, times 2) in the FIFO before requesting the Local Bus for reads. Nybble values 0h through Eh may be used. (Refer to Table 12.)<br>(15 - C0PLAF) > C0LPAE. | Yes | Yes | 0h | x |
| 11:8 | **DMA Channel 0 Local-to-PCI Almost Full (C0LPAF).** Number of full (Lword x 2) entries (plus 1, times 2) in the FIFO before requesting the PCI Bus for writes. Nybble values 0h through Eh may be used. (Refer to Table 12.) | Yes | Yes | 0h | x |
| 15:12 | **DMA Channel 0 PCI-to-Local Almost Empty (C0PLAE).** Number of empty (Lword x 2) entries (plus 1, times 2) in the FIFO before requesting the PCI Bus for reads. Nybble values 0h through Eh may be used. (Refer to Table 12.) | Yes | Yes | 0h | x |
| 19:16 | **DMA Channel 1 PCI-to-Local Almost Full (C1PLAF).** Number of full (Lword x 2) entries (plus 1, times 2) in the FIFO before requesting the Local Bus for writes. Nybble values 0h through Eh may be used. (Refer to Table 12.)<br>(15 - C1PLAF) > C1LPAE. | Yes | Yes | 0h | x |
| 23:20 | **DMA Channel 1 Local-to-PCI Almost Empty (C1LPAE).** Number of empty (Lword x 2) entries (plus 1, times 2) in the FIFO before requesting the Local Bus for reads. Nybble values 0h through Eh may be used. (Refer to Table 12.)<br>(15 - C1PLAF) > C1LPAE. | Yes | Yes | 0h | x |
| 27:24 | **DMA Channel 1 Local-to-PCI Almost Full (C1LPAF).** Number of full (Lword x 2) entries (plus 1, times 2) in the FIFO before requesting the PCI Bus for writes. Nybble values 0h through Eh may be used. (Refer to Table 12.) | Yes | Yes | 0h | x |
| 31:28 | **DMA Channel 1 PCI-to-Local Almost Empty (C1PLAE).** Number of empty (Lword x 2) entries (plus 1, times 2) sin the FIFO before requesting the PCI Bus for reads. Nybble values 0h through Eh may be used. (Refer to Table 12.) | Yes | Yes | 0h | x |

**Table 12: DMA Threshold Nybble Values**

| Nybble Value | Setting | Nybble Value | Setting | Nybble Value | Setting |
|---|---|---|---|---|---|
| 0h | 4 Lwords | 5h | 24 Lwords | Ah | 44 Lwords |
| 1h | 8 Lwords | 6h | 28 Lwords | Bh | 48 Lwords |
| 2h | 12 Lwords | 7h | 32 Lwords | Ch | 52 Lwords |
| 3h | 16 Lwords | 8h | 38 Lwords | Dh | 58 Lwords |
| 4h | 20 Lwords | 9h | 40 Lwords | Eh | 60 Lwords |

## DMADAn

DMA PCI Dual Address Cycle Upper Address

| Bit | Description | Read | Write | Value after Reset | Value to Use |
|---|---|---|---|---|---|
| 31:0 | **Upper 32 Bits of the PCI Dual Address Cycle PCI Address during DMA Cycles.** If set to 0h, the PCI 9056 performs a 32-bit address DMA access. | Yes | Yes | 0h | x |

## INTCSR

Interrupt Control/Status Register

| Bit | Description | Read | Write | Value after Reset | Value to Use |
|---|---|---|---|---|---|
| 0 | Writing 1 enables LSERR# to be asserted upon detection of a Local parity error or PCI Abort. | Yes | Yes | 0 | 0 |
| 1 | Writing 1 enables LSERR# to be asserted upon detection of an SERR# assertion in Host mode, or detection of a PCI parity error or a messaging queue outbound overflow. | Yes | Yes | 0 | 0 |
| 2 | **Generate PCI Bus SERR# Interrupt.** When set to 0, writing 1 asserts the PCI Bus SERR# interrupt. | Yes | Yes | 0 | 0 |
| 3 | **Mailbox Interrupt Enable.** Writing 1 enables a Local interrupt output (LINTo#) to be asserted when the PCI Bus writes to MBOX0 through MBOX3. To clear a LINTo# interrupt, the Local Bus Master must read the Mailbox. Used in conjunction with the Local Interrupt Output Enable bit (INTCSR[16]). | Yes | Yes | 0 | 0 |
| 4 | **Power Management Interrupt Enable.** Writing 1 enables a Local interrupt output (LINTo#) to be asserted when the Power Management Power State changes. | Yes | Yes | 0 | 0 |
| 5 | **Power Management Interrupt.** When set to 1, indicates a Power Management interrupt is pending. A Power Management interrupt is caused by a change in the Power Management Control/Status register Power State bits (PMCSR[1:0]). Writing 1 clears the interrupt. Writable from the PCI Bus only in the D0 power state. | Yes | Yes/Clr | 0 | 0 |

| | | | | | |
|---|---|---|---|---|---|
| 6 | **Direct Master Write/Direct Slave Read Local Data Parity Check Error Enable**. Writing 1 enables a Local Bus Data Parity Error signal to be asserted through the LSERR# pin. INTCSR[0] must be enabled for this to have an effect. | Yes | Yes | 0 | 0 |
| 7 | **Direct Master Write/Direct Slave Read Local Data Parity Check Error Status**. When set to 1, indicates the PCI 9056 has detected a Local data parity check error, even if Parity Check Error is disabled (INTCSR[6]=0). Writing 1 clears this bit to 0. | Yes | Yes/Clr | 0 | 0 |
| 8 | **PCI Interrupt Enable**. Writing 1 enables PCI interrupts (INTA#). | Yes | Yes | 1 | 1 |
| 9 | **PCI Doorbell Interrupt Enable.** Writing 1 enables Local-to-PCI Doorbell interrupts. Used in conjunction with the PCI Interrupt Enable bit (INTCSR[8]). Clearing the L2PDBELL register bits that caused the interrupt also clears the interrupt. | Yes | Yes | 0 | 0 |
| 10 | **PCI Abort Interrupt Enable.** Value of 1 enables a Master Abort or Master detection of a Target Abort to assert a PCI interrupt (INTA#). Used in conjunction with the PCI Interrupt Enable bit (INTCSR[8]). Clearing the Received Master and Target Abort bits (PCISR[13:12]) also clears the PCI interrupt. | Yes | Yes | 0 | 0 |
| 11 | **Local Interrupt Input Enable.** Writing 1 enables a Local interrupt input (LINTi#) assertion to assert a PCI interrupt (INTA#). Used in conjunction with the PCI Interrupt Enable bit (INTCSR[8]). De-asserting LINTi# also clears the interrupt. | Yes | Yes | 0 | 0 |
| 12 | **Retry Abort Enable.** Writing 1 enables the PCI 9056 to treat 256 consecutive Master Retries to a Target as a Target Abort. Writing 0 enables the PCI 9056 to attempt Master Retries indefinitely. | Yes | Yes | 0 | 0 |
| 13 | **PCI Doorbell Interrupt Active.** When set to 1, indicates the PCI Doorbell interrupt is active. | Yes | No | 0 | 0 |
| 14 | **PCI Abort Interrupt Active.** When set to 1, indicates the PCI Master or Target Abort interrupt is active. | Yes | No | 0 | 0 |
| 15 | **Local Interrupt Input Active.** When set to 1, indicates the Local interrupt input (LINTi#) is active. | Yes | No | 0 | 0 |
| 16 | **Local Interrupt Output Enable.** Writing 1 enables Local interrupt output (LINTo#). | Yes | Yes | 1 | 1 |
| 17 | **Local Doorbell Interrupt Enable.** Writing 1 enables PCI-to-Local Doorbell interrupts. Used in conjunction with the Local Interrupt Output Enable bit (INTCSR[16]). Clearing the P2LDBELL register bits that caused the interrupt also clears the interrupt. | Yes | Yes | 0 | 0 |

| | | | | | |
|---|---|---|---|---|---|
| 18 | **DMA Channel 0 Interrupt Enable.** Writing 1 enables DMA Channel 0 interrupts. Used in conjunction with the DMA Channel 0 Interrupt Select bit (DMAMODE0[17]). Setting the DMA Channel 0 Clear Interrupt bit (DMACSR0[3]=1) also clears the interrupt. | Yes | Yes | 0 | 0/1 |
| 19 | **DMA Channel 1 Interrupt Enable.** Writing 1 enables DMA Channel 1 interrupts. Used in conjunction with the DMA Channel 1 Interrupt Select bit (DMAMODE1[17]). Setting the DMA Channel 1 Clear Interrupt bit (DMACSR1[3]=1) also clears the interrupt. | Yes | Yes | 0 | 0/1 |
| 20 | **Local Doorbell Interrupt Active.** Reading 1 indicates the Local Doorbell interrupt is active. | Yes | No | 0 | 0 |
| 21 | **DMA Channel 0 Interrupt Active.** Reading 1 indicates the DMA Channel 0 interrupt is active. | Yes | No | 0 | 0 |
| 22 | **DMA Channel 1 Interrupt Active.** Reading 1 indicates the DMA Channel 1 interrupt is active. | Yes | No | 0 | 0 |
| 23 | **Built-In Self-Test (BIST) Interrupt Active.** Reading 1 indicates the BIST interrupt is active. The BIST interrupt is enabled by writing 1 to the PCI Built-In Self-Test Interrupt Enable bit (PCIBISTR[6]=1). Clearing the Enable bit (PCIBISTR[6]=0) also clears the interrupt. Note: Refer to the PCIBISTR register for a description of the self-test. | Yes | No | 0 | 0 |
| 24 | Reading 0 indicates the Direct Master was the Bus Master during a Master or Target Abort. | Yes | No | 1 | 1 |
| 25 | Reading 0 indicates that DMA Channel 0 was the Bus Master during a Master or Target Abort. | Yes | No | 1 | 1 |
| 26 | Reading 0 indicates that DMA Channel 1 was the Bus Master during a Master or Target Abort. | Yes | No | 1 | 1 |
| 27 | Reading 0 indicates that the PCI 9056 asserted a Target Abort after 256 consecutive Master Retries to a Target. | Yes | No | 1 | 1 |
| 28 | Reading 1 indicates that the PCI Bus wrote data to MBOX0. Enabled only if the Mailbox Interrupt Enable bit is set (INTCSR[3]=1). | Yes | No | 0 | 0 |
| 29 | Reading 1 indicates that the PCI Bus wrote data to MBOX1. Enabled only if the Mailbox Interrupt Enable bit is set (INTCSR[3]=1). | Yes | No | 0 | 0 |
| 30 | Reading 1 indicates that the PCI Bus wrote data to MBOX2. Enabled only if the Mailbox Interrupt Enable bit is set (INTCSR[3]=1). | Yes | No | 0 | 0 |

| | | | | | |
|---|---|---|---|---|---|
| 31 | Reading 1 indicates that the PCI Bus wrote data to MBOX3. Enabled only if the Mailbox Interrupt Enable bit is set (INTCSR[3]=1). | Yes | No | 0 | 0 |

# Additional Information

## *PLX PCI9056*

For more information about the PLX PCI9056 PCI Accelerator, contact PLX Technologies at:

www.plxtech.com

## *82C54 Timer/Counter Programming*

For more information about programming the MSM82C54 Timer/Counter Chips, contact Oki Semiconductor at:

www2.okisemi.com

## *Interrupt Programming*

For more information about interrupts and writing interrupt service routines, refer to the following book:

Interrupt-Driven PC System Design
by Joseph McGivern
ISBN: 0929392507

# DC Characteristics

## *Absolute Maximum Ratings*

| | Min | Max | Units |
|---|---|---|---|
| **DIO Vin[1]** | -0.5 | 5.5 | **V** |
| **Operating Temp** | -40 | +85 | **°C** |

## *DC Input / Output Levels*

| Input Standard | | | | Output Standard | | | |
|---|---|---|---|---|---|---|---|
| VIL | | VIH | | VOL | VOH | IOL | IOH |
| V, Min | V, Max | V, Min | V, Max | V, Max | V, Min | mA | mA |
| -0.5 | 0.8 | 2.0 | 5.5 | 0.4 | 2.4 | 24 | -24 |

1. DIO Vin DC overshoot must be limited to either 5.5V or 10mA and DC undershoot must be limited to either -0.5V or 10mA.
2. DIO pins may be driven to - 2.0V or + 7.0V provided these voltages last no longer than 11ns with a forcing current no greater than 100mA.
3. Inputs are terminated with 33Ω resistors and protection diodes.
4. DIO inputs should not be tied to voltages when the board is not powered.

# Limited Warranty

RTD Embedded Technologies, Inc. warrants the hardware and software products it manufactures and produces to be free from defects in materials and workmanship for one year following the date of shipment from RTD EMBEDDED TECHNOLOGIES, INC. This warranty is limited to the original purchaser of product and is not transferable.

During the one year warranty period, RTD EMBEDDED TECHNOLOGIES will repair or replace, at its option, any defective products or parts at no additional charge, provided that the product is returned, shipping prepaid, to RTD EMBEDDED TECHNOLOGIES. All replaced parts and products become the property of RTD EMBEDDED TECHNOLOGIES. Before returning any product for repair, customers are required to contact the factory for an RMA number.

THIS LIMITED WARRANTY DOES NOT EXTEND TO ANY PRODUCTS WHICH HAVE BEEN DAMAGED AS A RESULT OF ACCIDENT, MISUSE, ABUSE (such as: use of incorrect input voltages, improper or insufficient ventilation, failure to follow the operating instructions that are provided by RTD EMBEDDED TECHNOLOGIES, "acts of God" or other contingencies beyond the control of RTD EMBEDDED TECHNOLOGIES), OR AS A RESULT OF SERVICE OR MODIFICATION BY ANYONE OTHER THAN RTD EMBEDDED TECHNOLOGIES. EXCEPT AS EXPRESSLY SET FORTH ABOVE, NO OTHER WARRANTIES ARE EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, ANY IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, AND RTD EMBEDDED TECHNOLOGIES EXPRESSLY DISCLAIMS ALL WARRANTIES NOT STATED HEREIN. ALL IMPLIED WARRANTIES, INCLUDING IMPLIED WARRANTIES FOR MECHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, ARE LIMITED TO THE DURATION OF THIS WARRANTY. IN THE EVENT THE PRODUCT IS NOT FREE FROM DEFECTS AS WARRANTED ABOVE, THE PURCHASER'S SOLE REMEDY SHALL BE REPAIR OR REPLACEMENT AS PROVIDED ABOVE. UNDER NO CIRCUMSTANCES WILL RTD EMBEDDED TECHNOLOGIES BE LIABLE TO THE PURCHASER OR ANY USER FOR ANY DAMAGES, INCLUDING ANY INCIDENTAL OR CONSEQUENTIAL DAMAGES, EXPENSES, LOST PROFITS, LOST SAVINGS, OR OTHER DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PRODUCT.

SOME STATES DO NOT ALLOW THE EXCLUSION OR LIMITATION OF INCIDENTAL OR CONSEQUENTIAL DAMAGES FOR CONSUMER PRODUCTS AND SOME STATES DO NOT ALLOW LIMITATIONS ON HOW LONG AN IMPLIED WARRANTY LASTS, SO THE ABOVE LIMITATIONS OR EXCLUSIONS MAY NOT APPLY TO YOU.

THIS WARRANTY GIVES YOU SPECIFIC LEGAL RIGHTS, AND YOU MAY ALSO HAVE OTHER RIGHTS WHICH VARY FROM STATE TO STATE.